Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. Also, specific user environment and usage history can make it difficult to reproduce the problem. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Use of a static code analysis tool can help detect some possible problems. Whatever the approach to development may be, the final program must satisfy some fundamental properties. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Programming languages are essential for software development. It is usually easier to code in "high-level" languages than in "low-level" ones. 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). In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. Different programming languages support different styles of programming (called programming paradigms). 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. Computer programmers are those who write computer software. 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. Programs were mostly entered using punched cards or paper tape. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. 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. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages.