

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. However, readability is more than just programming style. Computer programmers are those who write computer software. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. However, readability is more than just programming style. Allen Downey, in his book *How To Think Like A Computer Scientist*, writes: Many computer languages provide a mechanism to call functions provided by shared libraries. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years. While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. There are many approaches to the Software development process. 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. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Techniques like Code refactoring can enhance readability. 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. However, readability is more than just programming style. Scripting and breakpointing is also part of this process. 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. These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics. While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se. For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software. Use of a static code analysis tool can help detect some possible problems. There exist a lot of different approaches for each of those tasks. Following a consistent programming style often helps readability.