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. Following a consistent programming style often helps readability. 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. There are many approaches to the Software development process. The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Many applications use a mix of several languages in their construction and use. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Programs were mostly entered using punched cards or paper tape. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. 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. There exist a lot of different approaches for each of those tasks. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Different programming languages support different styles of programming (called programming paradigms).