Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. However, readability is more than just programming style. One approach popular for requirements analysis is Use Case analysis. Computer programmers are those who write computer software. However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. 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. New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation). High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. 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. Scripting and breakpointing is also part of this process. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Different programming languages support different styles of programming (called programming paradigms). Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.