By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. 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. Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. One approach popular for requirements analysis is Use Case analysis. Following a consistent programming style often helps readability. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Programming languages are essential for software development. Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute. 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). Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Many applications use a mix of several languages in their construction and use. Techniques like Code refactoring can enhance readability. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. It is usually easier to code in "high-level" languages than in "low-level" ones. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). Use of a static code analysis tool can help detect some possible problems.