It affects the aspects of quality above, including portability, usability and most importantly maintainability. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). Debugging is often done with IDEs. Standalone debuggers like GDB are also used, and these often provide less of a visual environment, usually using a command line. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Programmable devices have existed for centuries. When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). Also, specific user environment and usage history can make it difficult to reproduce the problem. However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages. Programs were mostly entered using punched cards or paper tape. 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. 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). Various visual programming languages have also been developed with the intent to resolve readability concerns by adopting non-traditional approaches to code structure and display. 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. Many applications use a mix of several languages in their construction and use. 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Following a consistent programming style often helps readability. Ideally, the programming language best suited for the task at hand will be selected. 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. 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.