Auxiliary tasks accompanying and related to programming include analyzing requirements, testing, debugging (investigating and fixing problems), implementation of build systems, and management of derived artifacts, such as programs' machine code. It is usually easier to code in "high-level" languages than in "low-level" ones. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. It is usually easier to code in "high-level" languages than in "low-level" ones. Programs were mostly entered using punched cards or paper tape. 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). Techniques like Code refactoring can enhance readability. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. 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). 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Computer programmers are those who write computer software. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). It is very difficult to determine what are the most popular modern programming languages.