Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. One approach popular for requirements analysis is Use Case analysis. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. Different programming languages support different styles of programming (called programming paradigms). Use of a static code analysis tool can help detect some possible problems. 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. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. 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. It affects the aspects of quality above, including portability, usability and most importantly maintainability. Use of a static code analysis tool can help detect some possible problems. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. One approach popular for requirements analysis is Use Case analysis. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. 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). Also, specific user environment and usage history can make it difficult to reproduce the problem. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them.