Many applications use a mix of several languages in their construction and use. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Ideally, the programming language best suited for the task at hand will be selected. It is usually easier to code in "high-level" languages than in "low-level" ones. FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research. Different programming languages support different styles of programming (called programming paradigms). The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). 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. Programmable devices have existed for centuries. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Code-breaking algorithms have also existed for centuries. Also, specific user environment and usage history can make it difficult to reproduce the problem. It affects the aspects of quality above, including portability, usability and most importantly maintainability. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine. In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Unreadable code often leads to bugs, inefficiencies, and duplicated code. 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. 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.