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. Many applications use a mix of several languages in their construction and use. 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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Integrated development environments (IDEs) aim to integrate all such help. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. Techniques like Code refactoring can enhance readability. Code-breaking algorithms have also existed for centuries. Integrated development environments (IDEs) aim to integrate all such help. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Programming languages are essential for software development. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Also, specific user environment and usage history can make it difficult to reproduce the problem. It is usually easier to code in "high-level" languages than in "low-level" ones. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Integrated development environments (IDEs) aim to integrate all such help. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. 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. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.