They are the building blocks for all software, from the simplest applications to the most sophisticated ones. There are many approaches to the Software development process. It is usually easier to code in "high-level" languages than in "low-level" ones. Normally the first step in debugging is to attempt to reproduce the problem. Normally the first step in debugging is to attempt to reproduce the problem. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Code-breaking algorithms have also existed for centuries. Also, specific user environment and usage history can make it difficult to reproduce the problem. Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. 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. 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. 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. Whatever the approach to development may be, the final program must satisfy some fundamental properties. There exist a lot of different approaches for each of those tasks. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Following a consistent programming style often helps readability. 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. 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. Whatever the approach to development may be, the final program must satisfy some fundamental properties. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. 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. 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. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.