After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. It is very difficult to determine what are the most popular modern programming languages. Programming languages are essential for software development. Computer programmers are those who write computer software. Code-breaking algorithms have also existed for centuries. 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. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years. Code-breaking algorithms have also existed for centuries. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. Programmable devices have existed for centuries. 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, 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. Scripting and breakpointing is also part of this process. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. Many applications use a mix of several languages in their construction and use. 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. Code-breaking algorithms have also existed for centuries. 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).