There exist a lot of different approaches for each of those tasks..  
There are many approaches to the Software development process.  
 Different programming languages support different styles of programming (called programming paradigms).  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
 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.  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
 Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA).  
  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
Many applications use a mix of several languages in their construction and use.  
 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).  
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.  
 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.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.