There exist a lot of different approaches for each of those tasks.  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
Ideally, the programming language best suited for the task at hand will be selected.  
Programming languages are essential for software development.  
The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit.  
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.  
Also, those involved with software development may at times engage in reverse engineering, which is the practice of seeking to understand an existing program so as to re-implement its function in some way.  
Techniques like Code refactoring can enhance readability.  
Normally the first step in debugging is to attempt to reproduce the problem.  
Scripting and breakpointing is also part of this process.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.  
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.  
 The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).  
Proficient programming thus usually requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.