There exist a lot of different approaches for each of those tasks.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash.  
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.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams.  
For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software.  
Normally the first step in debugging is to attempt to reproduce the problem.  
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.  
Scripting and breakpointing is also part of this process.  
 Programs were mostly entered using punched cards or paper tape.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
 Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation.  
However, readability is more than just programming style.  
Use of a static code analysis tool can help detect some possible problems.