**Elementary programming in c**

**C** ([/ˈsiː/](https://en.wikipedia.org/wiki/Help%3AIPA_for_English), as in [the letter *c*](https://en.wikipedia.org/wiki/C)) is a [general-purpose](https://en.wikipedia.org/wiki/General-purpose_language), [imperative](https://en.wikipedia.org/wiki/Imperative_programming) computer [programming language](https://en.wikipedia.org/wiki/Programming_language), supporting [structured programming](https://en.wikipedia.org/wiki/Structured_programming), [lexical variable scope](https://en.wikipedia.org/wiki/Lexical_variable_scope) and [recursion](https://en.wikipedia.org/wiki/Recursion_%28computer_science%29), while a [static type system](https://en.wikipedia.org/wiki/Static_type_system) prevents many unintended operations. By design, C provides constructs that map efficiently to typical [machine instructions](https://en.wikipedia.org/wiki/Machine_instruction), and therefore it has found lasting use in applications that had formerly been coded in [assembly language](https://en.wikipedia.org/wiki/Assembly_language), including [operating systems](https://en.wikipedia.org/wiki/Operating_system), as well as various [application software](https://en.wikipedia.org/wiki/Application_software) for computers ranging from [supercomputers](https://en.wikipedia.org/wiki/Supercomputer) to [embedded systems](https://en.wikipedia.org/wiki/Embedded_system).

C was originally developed by [Dennis Ritchie](https://en.wikipedia.org/wiki/Dennis_Ritchie) between 1969 and 1973 at [Bell Labs](https://en.wikipedia.org/wiki/Bell_Labs),[[5]](https://en.wikipedia.org/wiki/C_%28programming_language%29#cite_note-chistory-5) and used to re-implement the [Unix](https://en.wikipedia.org/wiki/Unix) operating system.[[6]](https://en.wikipedia.org/wiki/C_%28programming_language%29#cite_note-AutoTX-1-6) It has since become one of the [most widely used programming languages](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity) of all time,[[7]](https://en.wikipedia.org/wiki/C_%28programming_language%29#cite_note-AutoTX-2-7)[[8]](https://en.wikipedia.org/wiki/C_%28programming_language%29#cite_note-AutoTX-3-8) with C [compilers](https://en.wikipedia.org/wiki/Compiler) from various vendors available for the majority of existing [computer architectures](https://en.wikipedia.org/wiki/Computer_architecture) and operating systems. C has been standardized by the [American National Standards Institute](https://en.wikipedia.org/wiki/American_National_Standards_Institute) (ANSI) since 1989 (see [ANSI C](https://en.wikipedia.org/wiki/ANSI_C)) and subsequently by the [International Organization for Standardization](https://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO).

**Design**

C is an [imperative](https://en.wikipedia.org/wiki/Imperative_programming) [procedural](https://en.wikipedia.org/wiki/Procedural_programming) language. It was designed to be compiled using a relatively straightforward [compiler](https://en.wikipedia.org/wiki/Compiler), to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal [run-time support](https://en.wikipedia.org/wiki/Run-time_system). Therefore, C was useful for many applications that had formerly been coded in assembly language, for example in [system programming](https://en.wikipedia.org/wiki/System_programming).

Despite its low-level capabilities, the language was designed to encourage [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) programming. A standards-compliant and [portably](https://en.wikipedia.org/wiki/Porting) written C program can be compiled for a very wide variety of computer platforms and operating systems with few changes to its source code. The language has become available on a very wide range of platforms, from embedded [microcontrollers](https://en.wikipedia.org/wiki/Microcontroller) to [supercomputers](https://en.wikipedia.org/wiki/Supercomputer).

**Overview**

Like most imperative languages in the [ALGOL](https://en.wikipedia.org/wiki/ALGOL) tradition, C has facilities for [structured programming](https://en.wikipedia.org/wiki/Structured_programming) and allows [lexical variable scope](https://en.wikipedia.org/wiki/Lexical_variable_scope) and recursion, while a static [type system](https://en.wikipedia.org/wiki/Type_system) prevents many unintended operations. In C, all [executable code](https://en.wikipedia.org/wiki/Executable_code) is contained within [subroutines](https://en.wikipedia.org/wiki/Subroutine), which are called "functions" (although not in the strict sense of [functional programming](https://en.wikipedia.org/wiki/Functional_programming)). [Function parameters](https://en.wikipedia.org/wiki/Function_parameter) are always passed by value. Pass-by-reference is simulated in C by explicitly passing [pointer](https://en.wikipedia.org/wiki/Pointer_%28computer_programming%29) values. C program source text is [free-format](https://en.wikipedia.org/wiki/Free-form_language), using the [semicolon](https://en.wikipedia.org/wiki/Semicolon) as a [statement](https://en.wikipedia.org/wiki/Statement_%28programming%29) terminator and [curly braces](https://en.wikipedia.org/wiki/Curly_braces) for grouping [blocks of statements](https://en.wikipedia.org/wiki/Blocks_of_statements).

The C language also exhibits the following characteristics:

* There is a small, fixed number of keywords, including a full set of [flow of control](https://en.wikipedia.org/wiki/Flow_of_control) primitives: [for](https://en.wikipedia.org/wiki/For_loop), [if/else](https://en.wikipedia.org/wiki/Conditional_%28programming%29), [while](https://en.wikipedia.org/wiki/While_loop), [switch](https://en.wikipedia.org/wiki/Switch_statement), and [do/while](https://en.wikipedia.org/wiki/Do_while_loop). User-defined names are not distinguished from keywords by any kind of [sigil](https://en.wikipedia.org/wiki/Sigil_%28computer_programming%29%22%20%5Co%20%22Sigil%20%28computer%20programming%29).
* There are a large number of arithmetical and logical operators, such as +, +=, ++, &, ~, etc.
* More than one [assignment](https://en.wikipedia.org/wiki/Assignment_%28computer_science%29) may be performed in a single statement.
* Function return values can be ignored when not needed.
* Typing is [static](https://en.wikipedia.org/wiki/Static_typing), but [weakly enforced](https://en.wikipedia.org/wiki/Strong_and_weak_typing): all data has a type, but implicit conversions may be performed.
* [Declaration](https://en.wikipedia.org/wiki/Declaration_%28computer_programming%29) [syntax](https://en.wikipedia.org/wiki/C_syntax) mimics usage context. C has no "define" keyword; instead, a statement beginning with the name of a type is taken as a declaration. There is no "function" keyword; instead, a function is indicated by the parentheses of an argument list.
* User-defined (typedef) and compound types are possible.
	+ Heterogeneous aggregate data types ([struct](https://en.wikipedia.org/wiki/Struct_%28C_programming_language%29%22%20%5Co%20%22Struct%20%28C%20programming%20language%29)) allow related data elements to be accessed and assigned as a unit.
	+ [Array](https://en.wikipedia.org/wiki/Array_data_type) indexing is a secondary notation, defined in terms of pointer arithmetic. Unlike structs, arrays are not first-class objects; they cannot be assigned or compared using single built-in operators. There is no "array" keyword, in use or definition; instead, square brackets indicate arrays syntactically, for example month[11].
	+ [Enumerated types](https://en.wikipedia.org/wiki/Enumerated_type) are possible with the enum keyword. They are not tagged, and are freely interconvertible with integers.
	+ [Strings](https://en.wikipedia.org/wiki/String_%28computer_science%29) are not a separate data type, but are conventionally implemented as [null-terminated](https://en.wikipedia.org/wiki/Null-terminated_string) arrays of characters.
* Low-level access to [computer memory](https://en.wikipedia.org/wiki/Computer_memory) is possible by converting machine addresses to typed [pointers](https://en.wikipedia.org/wiki/Pointer_%28computer_programming%29).
* [Procedures](https://en.wikipedia.org/wiki/Procedure_%28computer_science%29) (subroutines not returning values) are a special case of function, with an untyped return type void.
* Functions may not be defined within the lexical scope of other functions.
* Function and data pointers permit *ad hoc* [run-time polymorphism](https://en.wikipedia.org/wiki/Run-time_polymorphism).
* A [preprocessor](https://en.wikipedia.org/wiki/C_preprocessor%22%20%5Co%20%22C%20preprocessor) performs [macro](https://en.wikipedia.org/wiki/Macro_%28computer_science%29) definition, [source code](https://en.wikipedia.org/wiki/Source_code) file inclusion, and [conditional compilation](https://en.wikipedia.org/wiki/Conditional_compilation).
* There is a basic form of [modularity](https://en.wikipedia.org/wiki/Modular_programming): files can be compiled separately and [linked](https://en.wikipedia.org/wiki/Linker_%28computing%29) together, with control over which functions and data objects are visible to other files via [static](https://en.wikipedia.org/wiki/Static_%28keyword%29) and extern attributes.
* Complex functionality such as [I/O](https://en.wikipedia.org/wiki/Input/output), [string](https://en.wikipedia.org/wiki/String_%28computer_science%29) manipulation, and mathematical functions are consistently delegated to [library routines](https://en.wikipedia.org/wiki/Library_%28computing%29).

While C does not include some features found in some other languages, such as [object orientation](https://en.wikipedia.org/wiki/Object-oriented_programming) or [garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_%28computer_science%29), such features can be implemented or emulated in C, often by way of external libraries (e.g., the [Boehm garbage collector](https://en.wikipedia.org/wiki/Boehm_garbage_collector) or the [GLib Object System](https://en.wikipedia.org/wiki/GObject%22%20%5Co%20%22GObject)).