C là một ngôn ngữ mệnh lệnh được phát triển từ đầu thập niên 1970 bởi Dennis Ritchie để dùng trong hệ điều hành UNIX. Từ đó, ngôn ngữ này đã lan rộng ra nhiều hệ điều hành khác và trở thành một những ngôn ngữ phổ dụng nhất. C là ngôn ngữ rất có hiệu quả và được ưa chuộng nhất để viết các phần mềm hệ thống, mặc dù nó cũng được dùng cho việc viết các ứng dụng. Ngoài ra, C cũng thường được dùng làm phương tiện giảng dạy trong khoa học máy tính mặc dù ngôn ngữ này không được thiết kế dành cho người nhập môn.

Các đặc trưng[sửa | sửa mã nguồn]

Tổng quan[sửa | sửa mã nguồn]

C là một ngôn ngữ lập trình tương đối nhỏ gọn vận hành gần với phần cứng và nó giống với ngôn ngữ Assembler hơn hầu hết các ngôn ngữ bậc cao. Hơn thế, C đôi khi được đánh giá như là "có khả năng di động", cho thấy sự khác nhau quan trọng giữa nó với ngôn ngữ bậc thấp như là Assembler, đó là việc mã C có thể được dịch và thi hành trong hầu hết các máy tính, hơn hẳn các ngôn ngữ hiện tại trong khi đó thì Assembler chỉ có thể chạy trong một số máy tính đặc biệt. Vì lý do này C được xem là ngôn ngữ bậc trung.

C đã được tạo ra với một mục tiêu là làm cho nó thuận tiện để viết các chương trình lớn với số lỗi ít hơn trong mẫu hình lập trình thủ tục mà lại không đặt gánh nặng lên vai người viết ra trình dịch C, là những người bề bộn với các đặc tả phức tạp của ngôn ngữ. Cuối cùng C có thêm những chức năng sau:

Một ngôn ngữ cốt lõi đơn giản, với các chức năng quan trọng chẳng hạn như là những hàm hay việc xử lý tập tin sẽ được cung cấp bởi các bộ thư viện các thủ tục.

Tập trung trên mẫu hình lập trình thủ tục, với các phương tiện lập trình theo kiểu cấu trúc.

Một hệ thống kiểu đơn giản nhằm loại bỏ nhiều phép toán không có ý nghĩa thực dụng.

Dùng ngôn ngữ tiền xử lý, tức là các câu lệnh tiền xử lý C, cho các nhiệm vụ như là định nghĩa các macro và hàm chứa nhiều tập tin mã nguồn (bằng cách dùng câu lệnh tiền xử lý dạng #include chẳng hạn).

Mức thấp của ngôn ngữ cho phép dùng tới bộ nhớ máy tính qua việc sử dụng kiểu dữ liệu pointer.

Số lượng từ khóa rất nhỏ gọn.

Các tham số được đưa vào các hàm bằng giá trị, không bằng địa chỉ.

Hàm các con trỏ cho phép hình thành một nền tảng ban đầu cho tính đóng và tính đa hình.

Hỗ trợ các bản ghi hay các kiểu dữ liệu kết hợp do người dùng từ khóa định nghĩa struct cho phép các dữ liệu liên hệ nhau có thể được tập hợp lại và được điều chỉnh như là toàn bộ.

Một số chức năng khác mà C không có (hay còn thiếu) nhưng có thể tìm thấy ở các ngôn ngữ khác bao gồm:

An toàn kiểu,

Tự động Thu dọn rác,

Các lớp hay các đối tượng cùng với các ứng xử của chúng (xem thêm OOP),

Các hàm lồng nhau,

Lập trình tiêu bản hay Lập trình phổ dụng,

Quá tải và Quá tải toán tử,

Các hỗ trợ cho đa luồng, đa nhiệm và mạng.

Mặc dù C còn thiếu nhiều chức năng hữu ích nhưng lý do quan trọng để C được chấp nhận vì nó cho phép các trình dịch mới được tạo ra một cách nhanh chóng trên các nền tảng mới và vì nó cho phép người lập trình dễ kiểm soát được những gì mà chương trình (do họ viết) thực thi. Đây là điểm thường làm cho mã C chạy hiệu quả hơn các ngôn ngữ khác. Thường thì chỉ có ngôn ngữ ASM chỉnh bằng tay chạy nhanh hơn (ngôn ngữ C), bởi vì ASM kiểm soát được toàn bộ máy. Mặc dù vậy, với sự phát triển các trình dịch C, và với sự phức tạp của các CPU hiện đại có tốc độ cao, C đã dần thu nhỏ khác biệt về tốc độ này.

Một lý do nữa cho việc C được sử dụng rộng rãi và hiệu quả là do các trình dịch, các thư viện và các phần mềm thông dịch của các ngôn ngữ bậc cao khác lại thường được tạo nên từ C.

Ví dụ "hello, world"[sửa | sửa mã nguồn]

Ví dụ đơn giản sau đây được thấy trong lần in đầu tiên của cuốn "The C Programming Language", và đã trở thành bài tiêu chuẩn trong chương nhập môn của hầu hết các loại sách giáo khoa về lập trình. Chương trình hiển thị câu "hello, world!" trên đầu ra chuẩn, mà thường là một màn hình. Mặc dù vậy, nó có thể xuất ra một tập tin hay xuất ra trên một thiết bị phần cứng kể cả trên một vùng chứa, tùy thuộc vào việc đầu ra chuẩn được chỉ thị vào đâu khi chương trình này được thực thi.

Các kiểu

C có một hệ thống kiểu tương tự như của Pascal, mặc dù chúng khác nhau trong một số khía cạnh. Có nhiều kiểu cho các số nguyên với nhiều cỡ cho có dấu và không có dấu, có kiểu số floating point, kiểu các ký tự char, các kiểu thứ tự enum, kiểu bản ghi record và kiểu đơn vị union.

C tạo ra sự mở rộng mạnh mẽ việc sử dụng của kiểu các con trỏ pointer, một dạng đơn giản các tham chiếu mà chúng chứa địa chỉ các vùng nhớ. Các con trỏ có thể được tham chiếu ngược (dereference) để lấy về giá trị của dữ liệu được chứa trong địa chỉ đó (địa chỉ mà con trỏ chỉ vào). Địa chỉ này có thể được điều chỉnh bằng các phép gán thông thường và các phép toán số học trên con trỏ. Trong thời gian thực thi, một con trỏ đại diện cho một địa chỉ của bộ nhớ. Trong thời gian chuyển dịch, nó là một kiểu phức tạp đại diện cho cả địa chỉ và kiểu của dữ liệu. Điều này cho phép các biểu thức bao gồm các con trỏ được kiểm tra về kiểu. Các con trỏ thì được dùng cho nhiều mục tiêu trong C. Các dòng ký tự string thường được đại diện bởi một con trỏ chỉ tới một dãy của các ký tự. Sự cấp phát bộ nhớ động, được miêu tả sau đây, thì được tiến hành thông qua các con trỏ.

Một con trỏ rỗng có nghĩa là nó không chỉ đến một chỗ nào hết. Điều này có ích trong những trường hợp như là con trỏ next trong một nút cuối của một danh sách liên kết linked list. Việc tham chiếu ngược một con trỏ trống gây ra các biểu hiện không dự đoán trước được. Các con trỏ kiểu void thì lại có thể chỉ đến một đối tượng mà không cần biết kiểu của đối tượng đó. Điều này đặc biệt hữu dụng trong lập trình tiêu bản bởi vì cỡ và kiểu của các đối tượng mà chúng chỉ tới thì không thể biết được và do đó không thể thực hiện tham chiếu ngược, nhưng chúng lại có thể được hoán chuyển thành các con trỏ của các kiểu khác.

Các kiểu mảng array trong C thì có cỡ cố định, độ lớn tĩnh của nó phải được biết trước trong thời gian chuyển dịch. Điều này gây nhiều trở ngại trong thực tế bởi vì người ta có thể chỉ định các vùng nhớ ở thời gian thực thi dựa trên các thư viện chuẩn và hành xử chúng như là các mảng. Không như các ngôn ngữ khác, C biểu thị các mảng giống như trường hợp các con trỏ: chúng đóng vai trò một địa chỉ của bộ nhớ và một kiểu dữ liệu. Do đó, các giá trị chỉ số có thể vượt quá cỡ của một mảng.

C cũng cung cấp các kiểu mảng đa chiều. Các giá trị chỉ số của các mảng đa chiều thì được gán theo thứ tự hàng chính. Một cách có ý nghĩa thì các mảng này hoạt động như là mảng của các mảng nhưng thực chất chúng được phân bố như là mảng một chiều với việc tính và tạo các vị trí tương đối.

C thường được dùng trong việc lập trình các hệ thống bậc thấp, ở đó có thể cần thiết để xem số nguyên như là một địa chỉ của bộ nhớ, là một giá trị double precision, hay là một kiểu con trỏ. Trong các trường hợp này, C cung cấp việc hoán chuyển, mà phép toán này sẽ bắt buộc chuyển đổi giá trị từ một kiểu sang một kiểu khác. Dùng phép hoán chuyển sẽ làm mất đi phần nào tính an toàn mà thường được cung cấp bởi hệ thống kiểu.

Lưu trữ dữ liệu

Một trong những chức năng quan trọng nhất của một ngôn ngữ lập trình là việc cung cấp cơ sở cho việc quản lý bộ nhớ và các đối tượng được chứa trong bộ nhớ. C cung ứng 3 phương cách để cấp phát bộ nhớ cho các đối tượng:

Sự cấp phát vùng nhớ tĩnh: khoảng trống dành cho đối tượng thì được cung cấp trong phần mã nhị phân ở thời gian dịch; những đối tượng này có một thời gian sống lâu dài theo sự tồn tại của phần mã nhị phân chứa chúng (các đối tượng).

Sự cấp phát vùng nhớ tự động: Các đối tượng tạm thời có thể được chứa trong một chồng (stack), và khoảng trống này thì được trả về một cách tự động và có thể được dùng lại sau khi khối mã mà chúng (tức các đối tượng tạm thời) được khai báo đã thực thi xong.

Sự cấp phát vùng nhớ động: Các khối của bộ nhớ với bất kì cỡ lớn mong muốn nào đều có thể được yêu cầu (hay xin) trong thời gian thi hành bằng cách dùng các hàm thư viện như là malloc(), realloc() và free() từ một khu vực của bộ nhớ có tên là heap; các khối này có thể được tái dụng sau khi gọi hàm free() để hoàn trả chúng lại cho bộ nhớ.