In the C programming language, data types refers to an extensive system for declaring variables of different types. The language itself provides basic arithmetic types and syntax to build array and compound types. Several headers in the standard library contain definitions of support types, that have additional properties, such as exact size, guaranteed.

Basic types

The C language provides many basic types. Most of them are formed from one of the four basic arithmetic type identifiers in C (char, int, float and double), and optional specifiers (signed, unsigned, short, long). All available basic arithmetic types are listed below:

Type	Explanation	Type	Explanation
char	smallest addressable unit of the machine that can contain basic character set. It is an integer type. Actual type can be either signed or unsigned depending on implementation	signed char	same as char, but guaranteed to be signed.
		unsigned char	same as char, but guaranteed to be unsigned.
short short int signed short signed short int	*short* signed integer type. At least 16 bits in size.	unsigned short unsigned short int	same as short, but unsigned.
int signed int	basic signed integer type. At least 16 bits in size.	unsigned unsigned int	same as int, but unsigned.
long long int signed long signed long int	*long* signed integer type. At least 32 bits in size.	unsigned long unsigned long int	same as long, but unsigned.
long long long long int signed long long	*long long* signed integer type. At least 64 bits in size. Specified since	unsigned long long unsigned long long int	same as long long, but unsigned.

signed long long int	the C99 version of the standard.		Specified only inC99 version of the standard.
float	(single precision) floating-point type. Actual properties unspecified, however on most systems this is IEEE 754 single precision floating point format.		
double	double precision floating-point type. Actual properties unspecified, however on most systems this is IEEE 754 double precision floating point format.		
long double	extended precision floating-point type. Actual properties unspecified. Unlike types float and double, it can be either 80-bit floating point format, the non-IEEE "double-double" or IEEE 754 quadruple precision floating-point format if a higher precision format is provided, otherwise it is the same as double. See this page for details.		

The actual size of integer types varies by implementation. The only guarantee is that the long long is not smaller than long, which is not smaller than int, which is not smaller than short. Also, int should be the integer type that the target processor is most efficient working with. This allows great flexibility: for example, all types can be 64-bit. However, only several different integer width schemes (data models) are popular and since data model defines how different

programs communicate, a uniform data model is used within a given operating system application interface.

In practice it should be noted that char is usually 8 bits in size, short is usually 16 bits in size and long is usually 32 bits in size (likewise unsigned char, unsigned short and unsigned long). For example this holds true for platforms as diverse as 1990s Sun0S 4 Unix, Microsoft MSDOS, modern Linux, and Microchip MCC18 for embedded 8 bit PIC microcontrollers.

The actual size of floating point types also varies by implementation. The only guarantee is that the long double is not smaller than double, which is not smaller than float. Usually, 32-bit and 64-bit IEEE 754 floating point formats are used, if supported by hardware.

Boolean type

The boolean (true/false) type is _Bool. The stdbool.h type also defines a few useful identifiers as macros: bool is defined as _Bool, true as 1, false as 0. Additionally,__bool_true_false_are_defined is defined as 1. The _Bool type and stdbool.h header did not exist in pre-1999 versions of the standard.

Size and pointer difference types

The C language provides the separate types size_t and ptrdiff_t to represent memory-related quantities. Existing types were deemed insufficient, because their size is defined according to the target processor's arithmetic capabilities, not the memory capabilities, such as available address space. Both of these types are defined in the stddef.h header (cstddef header in C++).

size_t is used to represent the size of any object (including arrays) in the particular implementation. It is used as the return type of the sizeof operator. The maximum size of size_t is provided via SIZE_MAX, a macro constant which is defined in the stdint.h header (cstdint header in C++). It is guaranteed to be at least 65535.

ptrdiff_t is used to represent the difference between pointers.

Interface to the properties of the basic types

Information about the actual properties, such as size, of the basic arithmetic types, is provided via macro constants in two headers: limits.h header (climits header in C++) defines macros for integer types and float.h header (cfloat header in C++) defines macros for floating-point types. The actual values depend on the implementation.

Properties of integer types

- CHAR_BIT – size of the char type in bits (at least 8 bits)
- SCHAR_MIN, SHRT_MIN, INT_MIN, LONG_MIN, LLONG_MIN(C99) – minimum possible value of signed integer
  types: signed char, signed short, signed int, signed long,signed long long
- SCHAR_MAX, SHRT_MAX, INT_MAX, LONG_MAX, LLONG_MAX(C99) – maximum possible value of signed integer
  types: signed char, signed short, signed int, signed long,signed long long
- UCHAR_MAX, USHRT_MAX, UINT_MAX, ULONG_MAX, ULLONG_MAX(C99) – maximum possible value of unsigned integer
  types: unsigned char, unsigned short, unsigned int,unsigned long, unsigned long long
- CHAR_MIN – minimum possible value of char
- CHAR_MAX – maximum possible value of char
- MB_LEN_MAX – maximum number of bytes in a multibyte character

Properties of floating-point types

- FLT_MIN, DBL_MIN, LDBL_MIN – minimum value of float, double, long double respectively
- FLT_MAX, DBL_MAX, LDBL_MAX – maximum value of float, double, long double respectively
- FLT_ROUNDS – rounding mode for floating-point operations
- FLT_EVAL_METHOD – evaluation method of expressions involving different floating-point types (only available in C99)
- FLT_RADIX – radix of the exponent in the floating-point types
- FLT_DIG, DBL_DIG, LDBL_DIG – number of decimal digits that can be represented without losing precision by float, double, long double respectively
- FLT_EPSILON, DBL_EPSILON, LDBL_EPSILON – difference between 1.0 and the next representable value of float, double, long double respectively
- FLT_MANT_DIG, DBL_MANT_DIG, LDBL_MANT_DIG – number of FLT_RADIX-base digits in the floating-point mantissa for types float, double, long double respectively
- FLT_MIN_EXP, DBL_MIN_EXP, LDBL_MIN_EXP – minimum negative integer such that FLT_RADIX raised to a power one less than that number is a normalized float, double, longdouble respectively
- FLT_MIN_10_EXP, DBL_MIN_10_EXP, LDBL_MIN_10_EXP – minimum negative integer such that 10 raised to a power one less than that number is a normalized float, double, longdouble respectively
- FLT_MAX_EXP, DBL_MAX_EXP, LDBL_MAX_EXP – maximum positive integer such that FLT_RADIX raised to a power one more than that number is a normalized float, double, longdouble respectively
- FLT_MAX_10_EXP, DBL_MAX_10_EXP, LDBL_MAX_10_EXP – maximum positive integer such that 10 raised to a power one more than that number is a normalized float, double, longdouble respectively
- DECIMAL_DIG – minimum number of decimal digits needed to represent all the significant digits for long double.[4] The value is at least 10. (only available in C99)

Fixed width integer types

The C99 standard includes definitions of several new integer types to enhance the portability of programs.[2] The already available basic integer types were deemed insufficient, because their actual sizes are implementation defined and may vary across different systems. The new types are especially useful in embedded environments where hardware supports usually only several types and that support varies from system to system. All new types are defined in inttypes.h header (cinttypes header in C++) and also are available at stdint.h header (cstdintheader in C++). The types can be grouped into the following categories:

- Exact width integer types which are guaranteed to have the same number N of bits across all implementations. Included only if it is available in the implementation.
- Least width integer types which are guaranteed to be the smallest type available in the implementation, that has at least specified number N of bits. Guaranteed to be specified for at least N=8,16,32,64.
- Fastest integer types which are guaranteed to be the fastest integer type available in the implementation, that has at least specified number N of bits. Guaranteed to be specified for at least N=8,16,32,64.
- Pointer integer types which are guaranteed to be able to hold a pointer

- Maximum width integer types which are guaranteed to be the largest integer type in the implementation

The following table summarizes the types and the interface to acquire the implementation details (N refers to the number of bits):

Type category	Signed types			Unsigned types		
	Type	Minimum value	Maximum value	Type	Minimum value	Maximum value
Exact width	intN_t	INTN_MIN	INTN_MAX	uintN_t	0	UINTN_MAX
Least width	int_leastN_t	INT_LEASTN_MIN	INT_LEASTN_MAX	uint_leastN_t	0	UINT_LEASTN_MAX
Fastest	int_fastN_t	INT_FASTN_MIN	INT_FASTN_MAX	uint_fastN_t	0	UINT_FASTN_MAX
Pointer	intptr_t	INTPTR_MIN	INTPTR_MAX	uintptr_t	0	UINTPTR_MAX
Maximum width	intmax_t	INTMAX_MIN	INTMAX_MAX	uintmax_t	0	UINTMAX_MAX