Standard C Library Functions Table, By Name

Last Updated: 2023-04-11

This table briefly describes the C library functions, listed in alphabetical order. This table provides the include file name and the function prototype for each function.

Table 1. Standard C Library Functions

Function	System Include File	Function Prototype	Description
abort	stdlib.h	void abort(void);	Stops a program abnormally.
abs	stdlib.h	int abs(int *n*);	Calculates the absolute value of an integer argument *n*.
acos	math.h	double acos(double *x*);	Calculates the arc cosine of *x*.
asctime	time.h	char *asctime(const struct tm **time*);	Converts the *time* that is stored as a structure to a character string.
asctime_r	time.h	char *asctime_r (const struct tm *tm, char *buf);	Converts *tm* that is stored as a structure to a character string. (Restartable version of asctime.)
asin	math.h	double asin(double *x*);	Calculates the arc sine of *x*.
assert	assert.h	void assert(int *expression*);	Prints a diagnostic message and ends the program if the expression is false.
atan	math.h	double atan(double *x*);	Calculates the arc tangent of *x*.
atan2	math.h	double atan2(double *y*, double *x*);	Calculates the arc tangent of *y*/*x*.
atexit	stdlib.h	int atexit(void (**func*)(void));	Registers a function to be called at normal termination.
atof	stdlib.h	double atof(const char **string*);	Converts *string* to a double-precision floating-point value.

Function	**System Include File**	**Function Prototype**	**Description**
atoi	stdlib.h	int atoi(const char **string*);	Converts *string* to an integer.
atol	stdlib.h	long int atol(const char **string*);	Converts *string* to a long integer.
bsearch	stdlib.h	void *bsearch(const void **key*, const void **base*, size_t *num*, size_t *size*, int (**compare*) (const void **element1*, const void **element2*));	Performs a binary search on an array of *num* elements, each of *size* bytes. The array must be sorted in ascending order by the function pointed to by *compare*.
btowc	stdio.h wchar.h	wint_t btowc(int *c*);	Determines whether *c* constitues a valid multibyte character in the initial shift state.
calloc	stdlib.h	void *calloc(size_t *num*, size_t *size*);	Reserves storage space for an array of *num* elements, each of size *size*, and initializes the values of all elements to 0.
catclose[6]	nl_types.h	int catclose (nl_catd catd);	Closes a previously opened message catalog.
catgets[6]	nl_types.h	char *catgets(nl_catd catd, int set_id, int msg_id, const char *s);	Retrieves a message from an open message catalog.
catopen[6]	nl_types.h	nl_catd catopen (const char *name, int oflag);	Opens a message catalog, which must be done before a message can be retrieved.
ceil	math.h	double ceil(double *x*);	Calculates the double value representing the smallest integer that is greater than or equal to *x*.
clearerr	stdio.h	void clearerr(FILE **stream*);	Resets the error indicators and the end-of-file indicator for *stream*.
clock	time.h	clock_t clock(void);	Returns the processor time that has elapsed since the job was started.
cos	math.h	double cos(double *x*);	Calculates the cosine of *x*.
cosh	math.h	double cosh(double *x*);	Calculates the hyperbolic cosine of *x*.

Function	System Include File	Function Prototype	Description
ctime	time.h	char *ctime(const time_t **time*);	Converts *time* to a character string.
ctime64	time.h	char *ctime64(const time64_t **time*);	Converts *time* to a character string.
ctime_r	time.h	char *ctime_r(const time_t *time, char *buf);	Converts *time* to a character string. (Restartable version of ctime.)
ctime64_r	time.h	char *ctime64_r(const time64_t *time, char *buf);	Converts *time* to a character string. (Restartable version of ctime64.)
difftime	time.h	double difftime(time_t *time2*, time_t *time1*);	Computes the difference between *time2* and *time1*.
difftime64	time.h	double difftime64(time64_t *time2*, time64_t *time1*);	Computes the difference between *time2* and *time1*.
div	stdlib.h	div_t div(int *numerator*, int *denominator*);	Calculates the quotient and remainder of the division of *numerator* by *denominator*.
erf	math.h	double erf(double *x*);	Calculates the error function of *x*.
erfc	math.h	double erfc(double *x*);	Calculates the error function for large values of *x*.
exit	stdlib.h	void exit(int *status*);	Ends a program normally.
exp	math.h	double exp(double *x*);	Calculates the exponential function of a floating-point argument *x*.
fabs	math.h	double fabs(double *x*);	Calculates the absolute value of a floating-point argument *x*.
fclose	stdio.h	int fclose(FILE **stream*);	Closes the specified *stream*.
fdopen[5]	stdio.h	FILE *fdopen(int *handle*, const char **type*);	Associates an input or output stream with the file identified by handle.
feof	stdio.h	int feof(FILE **stream*);	Tests whether the end-of-file flag is set for a given *stream*.
ferror	stdio.h	int ferror(FILE **stream*);	Tests for an error indicator in reading from or writing

Function	System Include File	Function Prototype	Description
			to *stream*.
fflush[1]	stdio.h	int fflush(FILE **stream*);	Writes the contents of the buffer associated with the output *stream*.
fgetc[1]	stdio.h	int fgetc(FILE **stream*);	Reads a single unsigned character from the input *stream*.
fgetpos[1]	stdio.h	int fgetpos(FILE **stream*, fpos_t **pos*);	Stores the current position of the file pointer associated with *stream* into the object pointed to by *pos*.
fgets[1]	stdio.h	char *fgets(char **string*, int *n*, FILE **stream*);	Reads a string from the input *stream*.
fgetwc[6]	stdio.h wchar.h	wint_t fgetwc(FILE **stream*);	Reads the next multibyte character from the input stream pointed to by *stream*.
fgetws[6]	stdio.h wchar.h	wchar_t *fgetws(wchar_t **wcs*, int *n*, FILE **stream*);	Reads wide characters from the stream into the array pointed to by *wcs*.
fileno[5]	stdio.h	int fileno(FILE **stream*);	Determines the file handle currently associated with *stream*.
floor	math.h	double floor(double *x*);	Calculates the floating-point value representing the largest integer less than or equal to *x*.
fmod	math.h	double fmod(double *x*, double *y*);	Calculates the floating-point remainder of *x*/*y*.
fopen	stdio.h	FILE *fopen(const char **filename*, const char **mode*);	Opens the specified file.
fprintf	stdio.h	int fprintf(FILE **stream*, const char **format-string*, *arg-list*);	Formats and prints characters and values to the output *stream*.
fputc[1]	stdio.h	int fputc(int *c*, FILE **stream*);	Prints a character to the output *stream*.
fputs[1]	stdio.h	int fputs(const char **string*, FILE **stream*);	Copies a string to the output *stream*.
fputwc[6]	stdio.h wchar.h	wint_t fputwc(wchar_t *wc*, FILE **stream*);	Converts the wide character *wc* to a multibyte character and writes it to the output

Function	System Include File	Function Prototype	Description
			stream pointed to by *stream* at the current position.
fputws[6]	stdio.h wchar.h	int fputws(const wchar_t **wcs*, FILE **stream*);	Converts the wide-character string *wcs* to a multibyte-character string and writes it to *stream* as a multibyte character string.
fread	stdio.h	size_t fread(void **buffer*, size_t *size*, size_t *count*, FILE **stream*);	Reads up to *count* items of *size* length from the input *stream*, and stores them in *buffer*.
free	stdlib.h	void free(void **ptr*);	Frees a block of storage.
freopen	stdio.h	FILE *freopen(const char **filename*, const char **mode*, FILE **stream*);	Closes *stream*, and reassigns it to the file specified.
frexp	math.h	double frexp(double *x*, int **expptr*);	Separates a floating-point number into its mantissa and exponent.
fscanf	stdio.h	int fscanf(FILE **stream*, const char **format-string*, *arg-list*);	Reads data from *stream* into locations given by *arg-list*.
fseek[1]	stdio.h	int fseek(FILE **stream*, long int *offset*, int *origin*);	Changes the current file position associated with *stream* to a new location.
fsetpos[1]	stdio.h	int fsetpos(FILE **stream*, const fpos_t **pos*);	Moves the current file position to a new location determined by *pos*.
ftell[1]	stdio.h	long int ftell(FILE **stream*);	Gets the current position of the file pointer.
fwide[6]	stdio.h wchar.h	int fwide(FILE **stream*, int *mode*);	Determines the orientation of the stream pointed to by *stream*.
fwprintf[6]	stdio.h wchar.h	int fwprintf(FILE **stream*, const wchar_t **format*, *arg-list*);	Writes output to the stream pointed to by *stream*.
fwrite	stdio.h	size_t fwrite(const void **buffer*, size_t *size*,size_t *count*, FILE **stream*);	Writes up to *count* items of *size* length from *buffer* to *stream*.
fwscanf[6]	stdio.h wchar.h	int fwscanf(FILE **stream*, const wchar_t **format*, *arg-list*)	Reads input from the stream pointed to by

Function	System Include File	Function Prototype	Description
			stream.
gamma	math.h	double gamma(double *x*);	Computes the Gamma Function
getc[1]	stdio.h	int getc(FILE **stream*);	Reads a single character from the input *stream*.
getchar[1]	stdio.h	int getchar(void);	Reads a single character from *stdin.*
getenv	stdlib.h	char *getenv(const char **varname*);	Searches environment variables for *varname*.
gets	stdio.h	char *gets(char **buffer*);	Reads a string from *stdin*, and stores it in *buffer*.
getwc[6]	stdio.h wchar.h	wint_t getwc(FILE **stream*);	Reads the next multibyte character from *stream*, converts it to a wide character and advances the associated file position indicator for *stream*.
getwchar[6]	wchar.h	wint_t getwchar(void);	Reads the next multibyte character from stdin, converts it to a wide character, and advances the associated file position indicator for stdin.
gmtime	time.h	struct tm *gmtime(const time_t **time*);	Converts a *time* value to a structure of type tm.
gmtime64	time.h	struct tm *gmtime64(const time64_t **time*);	Converts a *time* value to a structure of type tm.
gmtime_r	time.h	struct tm *gmtime_r (const time_t *time, struct tm *result);	Converts a *time* value to a structure of type tm. (Restartable version of gmtime.)
gmtime64_r	time.h	struct tm *gmtime64_r (const time64_t *time, struct tm *result);	Converts a *time* value to a structure of type tm. (Restartable version of gmtime64.)
hypot	math.h	double hypot(double *side1*, double *side2*);	Calculates the hypotenuse of a right-angled triangle with sides of length *side1* and *side2*.
isalnum	ctype.h	int isalnum(int *c*);	Tests if *c* is alphanumeric.

Function	System Include File	Function Prototype	Description
isalpha	ctype.h	int isalpha(int *c*);	Tests if *c* is alphabetic.
isascii[4]	ctype.h	int isascii(int *c*);	Tests if *c* is within the 7-bit US-ASCII range.
isblank	ctype.h	int isblank(int *c*);	Tests if *c* is a blank or tab character.
iscntrl	ctype.h	int iscntrl(int *c*);	Tests if *c* is a control character.
isdigit	ctype.h	int isdigit(int *c*);	Tests if *c* is a decimal digit.
isgraph	ctype.h	int isgraph(int *c*);	Tests if *c* is a printable character excluding the space.
islower	ctype.h	int islower(int *c*);	Tests if *c* is a lowercase letter.
isprint	ctype.h	int isprint(int *c*);	Tests if *c* is a printable character including the space.
ispunct	ctype.h	int ispunct(int *c*);	Tests if *c* is a punctuation character.
isspace	ctype.h	int isspace(int *c*);	Tests if *c* is a whitespace character.
isupper	ctype.h	int isupper(int *c*);	Tests if *c* is an uppercase letter.
iswalnum[4]	wctype.h	int iswalnum (wint_t wc);	Checks for any alphanumeric wide character.
iswalpha[4]	wctype.h	int iswalpha (wint_t wc);	Checks for any alphabetic wide character.
iswblank[4]	wctype.h	int iswblank (wint_t wc);	Checks for any blank or tab wide character.
iswcntrl[4]	wctype.h	int iswcntrl (wint_t wc);	Tests for any control wide character.
iswctype[4]	wctype.h	int iswctype(wint_t wc, wctype_t wc_prop);	Determines whether or not the wide character wc has the property wc_prop.
iswdigit[4]	wctype.h	int iswdigit (wint_t wc);	Checks for any decimal-digit wide character.
iswgraph[4]	wctype.h	int iswgraph (wint_t wc);	Checks for any printing wide character except for the wide-character space.
iswlower[4]	wctype.h	int iswlower (wint_t wc);	Checks for any lowercase wide character.

Function	System Include File	Function Prototype	Description
iswprint[4]	wctype.h	int iswprint (wint_t wc);	Checks for any printing wide character.
iswpunct[4]	wctype.h	int iswpunct (wint_t wc);	Test for a wide non-alphanumeric, non-space character.
iswspace[4]	wctype.h	int iswspace (wint_t wc);	Checks for any wide character that corresponds to an implementation-defined set of wide characters for which iswalnum is false.
iswupper[4]	wctype.h	int iswupper (wint_t wc);	Checks for any uppercase wide character.
iswxdigit[4]	wctype.h	int iswxdigit (wint_t wc);	Checks for any hexadecimal digit character.
isxdigit[4]	wctype.h	int isxdigit(int *c*);	Tests if *c* is a hexadecimal digit.
j0	math.h	double j0(double *x*);	Calculates the Bessel function value of the first kind of order 0.
j1	math.h	double j1(double *x*);	Calculates the Bessel function value of the first kind of order 1.
jn	math.h	double jn(int *n*, double *x*);	Calculates the Bessel function value of the first kind of order *n*.
labs	stdlib.h	long int labs(long int *n*);	Calculates the absolute value of *n*.
ldexp	math.h	double ldexp(double *x*, int *exp*);	Returns the value of *x* multiplied by (2 to the power of *exp*).
ldiv	stdlib.h	ldiv_t ldiv(long int *numerator*, long int *denominator*);	Calculates the quotient and remainder of *numerator*/*denominator*.
localeconv	locale.h	struct lconv *localeconv(void);	Formats numeric quantities in struct lconv according to the current locale.
localtime	time.h	struct tm *localtime(const time_t **timeval*);	Converts *timeval* to a structure of type tm.
localtime64	time.h	struct tm *localtime64(const time64_t **timeval*);	Converts *timeval* to a structure of type tm.

Function	System Include File	Function Prototype	Description
localtime_r	time.h	struct tm *localtime_r (const time_t *timeval, struct tm *result);	Converts a *time* value to a structure of type *tm*. (Restartable version of localtime.)
localtime64_r	time.h	struct tm *localtime64_r (const time64_t *timeval, struct tm *result);	Converts a *time* value to a structure of type *tm*. (Restartable version of localtime64.)
log	math.h	double log(double *x*);	Calculates the natural logarithm of *x*.
log10	math.h	double log10(double *x*);	Calculates the base 10 logarithm of *x*.
longjmp	setjmp.h	void longjmp(jmp_buf *env*, int *value*);	Restores a stack environment previously set in *env* by the setjmp function.
malloc	stdlib.h	void *malloc(size_t *size*);	Reserves a block of storage.
mblen	stdlib.h	int mblen(const char **string*, size_t *n*);	Determines the length of a multibyte character *string*.
mbrlen[4]	wchar.h	int mbrlen (const char *s, size_t n, mbstate_t *ps);	Determines the length of a multibyte character. (Restartable version of mblen.)
mbrtowc[4]	wchar.h	int mbrtowc (wchar_t *pwc, const char *s, size_t n, mbstate_t *ps);	Convert a multibyte character to a wide character (Restartable version of mbtowc.)
mbsinit[4]	wchar.h	int mbsinit (const mbstate_t *ps);	Test state object **ps* for initial state.
mbsrtowcs[4]	wchar.h	size_t mbsrtowc (wchar_t *dst, const char **src, size_t len, mbstate_t *ps);	Convert multibyte string to a wide character string. (Restartable version of mbstowcs.)
mbstowcs	stdlib.h	size_t mbstowcs(wchar_t **pwc*, const char **string*, size_t *n*);	Converts the multibyte characters in *string* to their corresponding wchar_t codes, and stores not more than *n* codes in *pwc*.
mbtowc	stdlib.h	int mbtowc(wchar_t **pwc*, const char **string*, size_t *n*);	Stores the wchar_t code corresponding to the first *n* bytes of multibyte

Function	System Include File	Function Prototype	Description
			character *string* into the wchar_t character *pwc*.
memchr	string.h	void *memchr(const void **buf*, int *c*, size_t *count*);	Searches the first *count* bytes of *buf* for the first occurrence of *c* converted to an unsigned character.
memcmp	string.h	int memcmp(const void **buf1*, const void **buf2*, size_t *count*);	Compares up to *count* bytes of *buf1* and *buf2*.
memcpy	string.h	void *memcpy(void **dest*, const void **src*, size_t *count*);	Copies *count* bytes of *src* to *dest*.
memmove	string.h	void *memmove(void **dest*, const void **src*, size_t *count*);	Copies *count* bytes of *src* to *dest*. Allows copying between objects that overlap.
memset	string.h	void *memset(void **dest*, int *c*, size_t *count*);	Sets *count* bytes of *dest* to a value *c*.
mktime	time.h	time_t mktime(struct tm **time*);	Converts local *time* into calendar time.
mktime64	time.h	time64_t mktime64(struct tm **time*);	Converts local *time* into calendar time.
modf	math.h	double modf(double *x*, double **intptr*);	Breaks down the floating-point value *x* into fractional and integral parts.
nextafter	math.h	double nextafter(double *x*, double *y*);	Calculates the next representable value after *x* in the direction of *y*.
nextafterl	math.h	long double nextafterl(long double *x*, long double *y*);	Calculates the next representable value after *x* in the direction of *y*.
nexttoward	math.h	double nexttoward(double *x*, long double *y*);	Calculates the next representable value after *x* in the direction of *y*.
nexttowardl	math.h	long double nexttowardl(long double *x*, long double *y*);	Calculates the next representable value after *x* in the direction of *y*.
nl_langinfo[4]	langinfo.h	char *nl_langinfo(nl_item *item*);	Retrieve from the current locale the string that describes the requested information specified by *item*.
perror	stdio.h	void perror(const char **string*);	Prints an error message to stderr.

Function	System Include File	Function Prototype	Description
pow	math.h	double pow(double *x*, double *y*);	Calculates the value *x* to the power *y*.
printf	stdio.h	int printf(const char **format-string*, *arg-list*);	Formats and prints characters and values to stdout.
putc[1]	stdio.h	int putc(int *c*, FILE **stream*);	Prints *c* to the output *stream*.
putchar[1]	stdio.h	int putchar(int *c*);	Prints *c* to stdout.
putenv	stdlib.h	int *putenv(const char **varname*);	Sets the value of an environment variable by altering an existing variable or creating a new one.
puts	stdio.h	int puts(const char **string*);	Prints a string to stdout.
putwc[6]	stdio.h wchar.h	wint_t putwchar(wchar_t *wc*, FILE **stream*);	Converts the wide character *wc* to a multibyte character, and writes it to the stream at the current position.
putwchar[6]	wchar.h	wint_t putwchar(wchar_t *wc*);	Converts the wide character *wc* to a multibyte character and writes it to stdout.
qsort	stdlib.h	void qsort(void **base*, size_t *num*, size_t *width*, int(**compare*)(const void **element1*, const void **element2*));	Performs a quick sort of an array of *num* elements, each of *width* bytes in size.
quantexpd32	math.h	_Decimal32 quantized32(_Decimal32 *x*, _Decimal32 *y*);	Compute the quantum exponent of a single-precision decimal floating-point value.
quantexpd64	math.h	_Decimal64 quantized64(_Decimal64 *x*, _Decimal64 *y*);	Compute the quantum exponent of a double-precision decimal floating-point value.
quantexpd128	math.h	_Decimal128 quantized128(_Decimal128 *x*, _Decimal128 *y*);	Compute the quantum exponent of a quad-precision decimal floating-point value.
quantized32	math.h	int quantexpd32(_Decimal32 *x*);	Set the quantum exponent of a single-precision decimal floating-point

Function	System Include File	Function Prototype	Description
			value to the quantum exponent of another single-precision decimal floating-point value.
quantized64	math.h	int quantexpd64(_Decimal64 *x*);	Set the quantum exponent of a double-precision decimal floating-point value to the quantum exponent of another double-precision decimal floating-point value.
quantized128	math.h	int quantexpd128(_Decimal128 *x*);	Set the quantum exponent of a quad-precision decimal floating-point value to the quantum exponent of another quad-precision decimal floating-point value.
samequantumd32	math.h	__bool__ samequantumd32(_Decimal32 *x*, _Decimal32 *y*);	Determine if the quantum exponents of two single-precision decimal floating-point values are the same.
samequantumd64	math.h	__bool__ samequantumd64(_Decimal64 *x*, _Decimal64 *y*);	Determine if the quantum exponents of two double-precision decimal floating-point values are the same.
samequantumd128	math.h	__bool__ samequantumd128(_Decimal128 *x*, _Decimal128 *y*);	Determine if the quantum exponents of two quad-precision decimal floating-point values are the same.
raise	signal.h	int raise(int *sig*);	Sends the signal *sig* to the running program.
rand	stdlib.h	int rand(void);	Returns a pseudo-random integer.
rand_r	stdlib.h	int rand_r(void);	Returns a pseudo-random integer. (Restartable version)
realloc	stdlib.h	void *realloc(void *ptr*, size_t *size*);	Changes the *size* of a previously reserved storage block.
regcomp	regex.h	int regcomp(regex_t *preg*, const char *pattern*, int *cflags*);	Compiles the source regular expression pointed to by *pattern* into an executable version and

Function	System Include File	Function Prototype	Description
			stores it in the location pointed to by *preg*.
regerror	regex.h	size_t regerror(int *errcode*, const regex_t **preg*, char **errbuf*, size_t *errbuf_size*);	Finds the description for the error code *errcode* for the regular expression *preg*.
regexec	regex.h	int regexec(const regex_t **preg*, const char **string*, size_t *nmatch*, regmatch_t **pmatch*, int *eflags*);	Compares the null-ended string *string* against the compiled regular expression *preg* to find a match between the two.
regfree	regex.h	void regfree(regex_t *preg);	Frees any memory that was allocated by regcomp to implement the regular expression *preg*.
remove	stdio.h	int remove(const char **filename*);	Deletes the file specified by *filename*.
rename	stdio.h	int rename(const char **oldname*, const char **newname*);	Renames the specified file.
rewind[1]	stdio.h	void rewind(FILE **stream*);	Repositions the file pointer associated with *stream* to the beginning of the file.
scanf	stdio.h	int scanf(const char **format-string*, *arg-list*);	Reads data from stdin into locations given by *arg-list*.
setbuf	stdio.h	void setbuf(FILE **stream*, char **buffer*);	Controls buffering for *stream*.
setjmp	setjmp.h	int setjmp(jmp_buf *env*);	Saves a stack environment that can be subsequently restored by longjmp.
setlocale	locale.h	char *setlocale(int *category*, const char **locale*);	Changes or queries variables defined in the *locale*.
setvbuf	stdio.h	int setvbuf(FILE **stream*, char **buf*, int *type*, size_t *size*);	Controls buffering and buffer *size* for *stream*.
signal	signal.h	void(*signal (int *sig*, void(**func*)(int))) (int);	Registers func as a signal handler for the signal sig.
sin	math.h	double sin(double *x*);	Calculates the sine of *x*.
sinh	math.h	double sinh(double *x*);	Calculates the hyperbolic sine of *x*.
snprintf	stdio.h	int snprintf(char *outbuf, size_t n, const char*, ...)	Same as sprintf except that the function will stop

Function	System Include File	Function Prototype	Description
			after n characters have been written to outbuf.
sprintf	stdio.h	int sprintf(char **buffer*, const char **format-string*, *arg-list*);	Formats and stores characters and values in *buffer*.
sqrt	math.h	double sqrt(double *x*);	Calculates the square root of *x*.
srand	stdlib.h	void srand(unsigned int *seed*);	Sets the *seed* for the pseudo-random number generator.
sscanf	stdio.h	int sscanf(const char **buffer*, const char **format*, *arg-list*);	Reads data from *buffer* into the locations given by *arg-list*.
strcasecmp	strings.h	int srtcasecmp(const char *string1, const char *string2);	Compares strings without case sensitivity.
strcat	string.h	char *strcat(char **string1*, const char **string2*);	Concatenates *string2* to *string1*.
strchr	string.h	char *strchr(const char **string*, int *c*);	Locates the first occurrence of *c* in *string*.
strcmp	string.h	int strcmp(const char **string1*, const char **string2*);	Compares the value of *string1* to *string2*.
strcoll	string.h	int strcoll(const char **string1*, const char **string2*);	Compares two strings using the collating sequence in the current locale.
strcpy	string.h	char *strcpy(char **string1*, const char **string2*);	Copies *string2* into *string1*.
strcspn	string.h	size_t strcspn(const char **string1*, const char **string2*);	Returns the length of the initial substring of *string1* consisting of characters not contained in *string2*.
strerror	string.h	char *strerror(int *errnum*);	Maps the error number in *errnum* to an error message string.
strfmon[4]	wchar.h	int strfmon (char *s, size_t maxsize, const char *format, ...);	Converts monetary value to string.
strftime	time.h	size_t strftime (char **dest*, size_t *maxsize*, const char **format*, const struct tm **timeptr*);	Stores characters in an array pointed to by *dest*, according to the string determined by *format*.
strlen	string.h	size_t strlen(const char **string*);	Calculates the length of *string*.

Function	System Include File	Function Prototype	Description
strncasecmp	strings.h	int strncasecmp(const char *string1, const char *string2, size_t count);	Compares strings without case sensitivity.
strncat	string.h	char *strncat(char *_string1_, const char *_string2_, size_t _count_);	Concatenates up to _count_ characters of _string2_ to _string1_.
strncmp	string.h	int strncmp(const char *_string1_, const char *_string2_, size_t _count_);	Compares up to _count_ characters of _string1_ and _string2_.
strncpy	string.h	char *strncpy(char *_string1_, const char *_string2_, size_t _count_);	Copies up to _count_ characters of _string2_ to _string1_.
strpbrk	string.h	char *strpbrk(const char *_string1_, const char *_string2_);	Locates the first occurrence in _string1_ of any character in _string2_.
strptime[4]	time.h	char *strptime (const char *buf, const char *format, struct tm *tm);	Date and time conversion
strrchr	string.h	char *strrchr(const char *_string_, int _c_);	Locates the last occurrence of _c_ in _string_.
strspn	string.h	size_t strspn(const char *_string1_, const char *_string2_);	Returns the length of the initial substring of _string1_ consisting of characters contained in _string2_.
strstr	string.h	char *strstr(const char *_string1_, const char *_string2_);	Returns a pointer to the first occurrence of _string2_ in _string1_.
strtod	stdlib.h	double strtod(const char *_nptr_, char **_endptr_);	Converts _nptr_ to a double precision value.
strtod32	stdlib.h	_Decimal32 strtod32(const char *_nptr_, char **_endptr_);	Converts _nptr_ to a single-precision decimal floating-point value.
strtod64	stdlib.h	_Decimal64 strtod64(const char *_nptr_, char **_endptr_);	Converts _nptr_ to a double-precision decimal floating-point value.
strtod128	stdlib.h	_Decimal128 strtod128(const char *_nptr_, char **_endptr_);	Converts _nptr_ to a quad-precision decimal floating-point value.
strtof	stdlib.h	float strtof(const char *_nptr_, char **_endptr_);	Converts _nptr_ to a float value.
strtok	string.h	char *strtok(char *_string1_, const char *_string2_);	Locates the next token in _string1_ delimited by the next character in _string2_.

Function	System Include File	Function Prototype	Description
strtok_r	string.h	char *strtok_r(char **string*, const char **seps*, char ***lasts*);	Locates the next token in *string* delimited by the next character in *seps*. (Restartable version of strtok.)
strtol	stdlib.h	long int strtol(const char **nptr*, char ***endptr*, int *base*);	Converts *nptr* to a signed long integer.
strtold	stdlib.h	long double strtold(const char **nptr*, char ***endptr*);	Converts *nptr* to a long double value.
strtoul	stdlib.h	unsigned long int strtoul(const char **string1*, char ***string2*, int *base*);	Converts *string1* to an unsigned long integer.
strxfrm	string.h	size_t strxfrm(char **string1*, const char **string2*, size_t *count*);	Converts *string2* and places the result in *string1*. The conversion is determined by the program's current locale.
swprintf	wchar.h	int swprintf(wchar_t **wcsbuffer*, size_t *n*, const wchar_t **format*, *arg-list*);	Formats and stores a series of wide characters and values into the wide-character buffer *wcsbuffer*.
swscanf	wchar.h	int swscanf (const wchar_t **buffer*, const wchar_t **format*, *arg-list*)	Reads data from *buffer* into the locations given by *arg-list*.
system	stdlib.h	int system(const char **string*);	Passes *string* to the system command analyzer.
tan	math.h	double tan(double *x*);	Calculates the tangent of *x*.
tanh	math.h	double tanh(double *x*);	Calculates the hyperbolic tangent of *x*.
time	time.h	time_t time(time_t **timeptr*);	Returns the current calendar time.
time64	time.h	time64_t time64(time64_t **timeptr*);	Returns the current calendar time.
tmpfile	stdio.h	FILE *tmpfile(void);	Creates a temporary binary file and opens it.
tmpnam	stdio.h	char *tmpnam(char **string*);	Generates a temporary file name.
toascii	ctype.h	int toascii(int *c*);	Converts *c* to a character in the 7-bit US-ASCII character set.

Function	System Include File	Function Prototype	Description
tolower	ctype.h	int tolower(int *c*);	Converts *c* to lowercase.
toupper	ctype.h	int toupper(int *c*);	Converts *c* to uppercase.
towctrans	wctype.h	wint_t towctrans(wint_t *wc*, wctrans_t *desc*);	Translates the wide character *wc* based on the mapping described by *desc*.
towlower[4]	wctype.h	wint_t towlower (wint_t wc);	Converts uppercase letter to lowercase letter.
towupper[4]	wctype.h	wint_t towupper (wint_t wc);	Converts lowercase letter to uppercase letter.
ungetc[1]	stdio.h	int ungetc(int *c*, FILE **stream*);	Pushes *c* back onto the input *stream*.
ungetwc[6]	stdio.h wchar.h	wint_t ungetwc(wint_t *wc*, FILE **stream*);	Pushes the wide character *wc* back onto the input stream.
va_arg	stdarg.h	*var_type* va_arg(va_list *arg_ptr*, var_type);	Returns the value of one argument and modifies *arg_ptr* to point to the next argument.
va_copy	stdarg.h	void *va_copy*(va_list *dest*, va_list *src*);	Initializes *dest* as a copy of *src*.
va_end	stdarg.h	void *va_end*(va_list *arg_ptr*);	Facilitates normal return from variable argument list processing.
va_start	stdarg.h	void *va_start*(va_list *arg_ptr, variable_name*);	Initializes *arg_ptr* for subsequent use by *va_arg* and *va_end*.
vfprintf	stdio.h stdarg.h	int vfprintf(FILE **stream*, const char **format*, *va_list arg_ptr*);	Formats and prints characters to the output *stream* using a variable number of arguments.
vfscanf	stdio.h stdarg.h	int vfscanf(FILE **stream*, const char **format*, va_list *arg_ptr*);	Reads data from a specified stream into locations given by a variable number of arguments.
vfwprintf[6]	stdarg.h stdio.h wchar.h	int vfwprintf(FILE **stream*, const wchar_t **format*, *va_list arg*);	Equivalent to fwprintf, except that the variable argument list is replaced by *arg*.

Function	System Include File	Function Prototype	Description
vfwscanf	stdio.h stdarg.h	int vfwscanf(FILE *stream*, const wchar_t **format*, *va_list arg_ptr*);	Reads wide data from a specified stream into locations given by a variable number of arguments.
vprintf	stdio.h stdarg.h	int vprintf(const char **format*, va_list *arg_ptr*);	Formats and prints characters to stdout using a variable number of arguments.
vscanf	stdio.h stdarg.h	int vscanf(const char **format*, *va_list arg_ptr*);	Reads data from stdin into locations given by a variable number of arguments.
vsprintf	stdio.h stdarg.h	int vsprintf(char **target-string*, const char **format*, va_list *arg_ptr*);	Formats and stores characters in a buffer using a variable number of arguments.
vsnprintf	stdio.h	int vsnprintf(char *outbuf, size_t n, const char*, va_list);	Same as vsprintf except that the function will stop after n characters have been written to outbuf.
vsscanf	stdio.h stdarg.h	int vsscanf(const char**buffer*, const char **format*, *va_list arg_ptr*);	Reads data from a buffer into locations given by a variable number of arguments.
vswprintf	stdarg.h wchar.h	int vswprintf(wchar_t **wcsbuffer*, size_t *n*, const wchar_t **format*, *va_list arg*);	Formats and stores a series of wide characters and values in the buffer *wcsbuffer*.
vswscanf	stdio.h wchar.h	int vswscanf(const wchar_t **buffer*, const wchar_t **format*, *va_list arg_ptr*);	Reads wide data from a buffer into locations given by a variable number of arguments.
vwprintf[6]	stdarg.h wchar.h	int vwprintf(const wchar_t **format*, *va_list arg*);	Equivalent to wprintf, except that the variable argument list is replaced by *arg*.
vwscanf	stdio.h wchar.h	int vwscanf(const wchar_t **format*, *va_list arg_ptr*);	Reads wide data from stdin into locations given by a variable number of arguments.
wcrtomb[4]	wchar.h	int wcrtomb (char *s, wchar_t wchar, mbstate_t *pss);	Converts a wide character to a multibyte character.

Function	System Include File	Function Prototype	Description
			(Restartable version of wctomb.)
wcscat	wchar.h	wchar_t *wcscat(wchar_t *string1*, const wchar_t *string2*);	Appends a copy of the string pointed to by *string2* to the end of the string pointed to by *string1*.
wcschr	wchar.h	wchar_t *wcschr(const wchar_t *string*, wchar_t *character*);	Searches the wide-character string pointed to by *string* for the occurrence of *character*.
wcscmp	wchar.h	int wcscmp(const wchar_t *string1*, const wchar_t *string2*);	Compares two wide-character strings, *string1* and *string2*.
wcscoll[4]	wchar.h	int wcscoll (const wchar_t *wcs1, const wchar_t *wcs2);	Compares two wide-character strings using the collating sequence in the current locale.
wcscpy	wchar.h	wchar_t *wcscpy(wchar_t *string1*, const wchar_t *string2*);	Copies the contents of *string2* (including the ending wchar_t null character) into *string1*.
wcscspn	wchar.h	size_t wcscspn(const wchar_t *string1*, const wchar_t *string2*);	Determines the number of wchar_t characters in the initial segment of the string pointed to by *string1* that do not appear in the string pointed to by *string2*.
wcsftime	wchar.h	size_t wcsftime(wchar_t *wdest*, size_t *maxsize*, const wchar_t *format*, const struct tm *timeptr*);	Converts the time and date specification in the *timeptr* structure into a wide-character string.
wcslen	wchar.h	size_t wcslen(const wchar_t *string*);	Computes the number of wide-characters in the string pointed to by *string*.
wcslocaleconv	locale.h	struct wcslconv *wcslocaleconv(void);	Formats numeric quantities in struct wcslconv according to the current locale.
wcsncat	wchar.h	wchar_t *wcsncat(wchar_t *string1*, const wchar_t *string2*, size_t *count*);	Appends up to *count* wide characters from *string2* to the end of *string1*, and

Function	System Include File	Function Prototype	Description
			appends a wchar_t null character to the result.
wcsncmp	wchar.h	int wcsncmp(const wchar_t **string1*, const wchar_t **string2*, size_t *count*);	Compares up to *count* wide characters in *string1* to *string2*.
wcsncpy	wchar.h	wchar_t *wcsncpy(wchar_t **string1*, const wchar_t **string2*, size_t *count*);	Copies up to *count* wide characters from *string2* to *string1*.
wcspbrk	wchar.h	wchar_t *wcspbrk(const wchar_t **string1*, const wchar_t **string2*);	Locates the first occurrence in the string pointed to by *string1* of any wide characters from the string pointed to by *string2*.
wcsptime	wchar.h	wchar_t *wcsptime (const wchar_t *buf, const wchar_t *format, struct tm *tm);	Date and time conversion. Equivalent to strptime(), except that it uses wide characters.
wcsrchr	wchar.h	wchar_t *wcsrchr(const wchar_t **string*, wchar_t *character*);	Locates the last occurrence of *character* in the string pointed to by *string*.
wcsrtombs[4]	wchar.h	size_t wcsrtombs (char *dst, const wchar_t **src, size_t len, mbstate_t *ps);	Converts wide character string to multibyte string. (Restartable version of wcstombs.)
wcsspn	wchar.h	size_t wcsspn(const wchar_t **string1*, const wchar_t **string2*);	Computes the number of wide characters in the initial segment of the string pointed to by *string1*, which consists entirely of wide characters from the string pointed to by *string2*.
wcsstr	wchar.h	wchar_t *wcsstr(const wchar_t **wcs1*, const wchar_t **wcs2*);	Locates the first occurrence of *wcs2* in *wcs1*.
wcstod	wchar.h	double wcstod(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a double value.
wcstod32	wchar.h	_Decimal32 wcstod32(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to

Function	System Include File	Function Prototype	Description
			a single-precision decimal floating-point value.
wcstod64	wchar.h	_Decimal64 wcstod64(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a double-precision decimal floating-point value.
wcstod128	wchar.h	_Decimal128 wcstod128(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a quad-precision decimal floating-point value.
wcstof	wchar.h	float wcstof(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a float value.
wcstok	wchar.h	wchar_t *wcstok(wchar_t **wcs1*, const wchar_t **wcs2*, wchar_t ***ptr*)	Breaks *wcs1* into a sequence of tokens, each of which is delimited by a wide character from the wide string pointed to by *wcs2*.
wcstol	wchar.h	long int wcstol(const wchar_t **nptr*, wchar_t ***endptr*, int *base*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a long integer value.
wcstold	wchar.h	long double wcstold(const wchar_t **nptr*, wchar_t ***endptr*);	Converts the initial portion of the wide-character string pointed to by *nptr* to a long double value.
wcstombs	stdlib.h	size_t wcstombs(char *dest, const wchar_t **string*, size_t *count*);	Converts the wchar_t *string* into a multibyte string *dest*.
wcstoul	wchar.h	unsigned long int wcstoul(const wchar_t **nptr*, wchar_t ***endptr*, int *base*);	Converts the initial portion of the wide-character string pointed to by *nptr* to an unsigned long integer value.
wcsxfrm[4]	wchar.h	size_t wcsxfrm (wchar_t **wcs1*, const wchar_t **wcs2*, size_t *n*);	Transforms a wide-character string to values which represent character collating weights and places the resulting wide-

Function	System Include File	Function Prototype	Description
			character string into an array.
wctob	stdarg.h wchar.h	int wctob(wint_t *wc*);	Determines whether *wc* corresponds to a member of the extended character set whose multibyte character representation is a single byte when in the initial shift state.
wctomb	stdlib.h	int wctomb(char **string*, wchar_t *character*);	Converts the wchar_t value of *character* into a multibyte *string*.
wctrans	wctype.h	wctrans_t wctrans(const char **property*);	Constructs a value with type wctrans_t that describes a mapping between wide characters identified by the string argument property.
wctype[4]	wchar.h	wctype_t wctype (const char **property*);	Obtains handle for character property classification.
wcwidth	wchar.h	int wcswidth(const wchar_t **pwcs*, size_t *n*);	Determine the display width of a wide character string.
wmemchr	wchar.h	wchar_t *wmemchr(const wchar_t **s*, wchar_t *c*, size_t *n*);	Locates the first occurrence of *c* in the initial *n* wide characters of the object pointed to by *s.*
wmemcmp	wchar.h	int wmemcmp(const wchar_t **s1*, const wchar_t **s2*, size_t *n*);	Compares the first *n* wide characters of the object pointed to by *s1* to the first *n* characters of the object pointed to by *s2*.
wmemcpy	wchar.h	wchar_t *wmemcpy(wchar_t **s1*, const wchar_t **s2*, size_t *n*);	Copies *n* wide characters from the object pointed to by *s2* to the object pointed to by *s1*.
wmemmove	wchar.h	wchar_t *wmemmove(wchar_t **s1*, const wchar_t **s2*, size_t *n*);	Copies *n* wide characters from the object pointed to by *s2* to the object pointed to by *s1*.
wmemset	wchar.h	wchar_t *wmemset(wchar_t **s*, wchar_t *c*, size_t *n*);	Copies the value of *c* into each of the first *n* wide

Function	System Include File	Function Prototype	Description
			characters of the object pointed to by *s*.
wprintf[6]	wchar.h	int wprintf(const wchar_t **format*, *arg-list*);	Equivalent to fwprintf with the argument stdout interposed before the arguments to wprintf.
wscanf[6]	wchar.h	int wscanf(const wchar_t **format*, *arg-list*);	Equivalent to fwscanf with the argument stdin interposed before the arguments of wscanf.
y0	math.h	double y0(double *x*);	Calculates the Bessel function value of the second kind of order 0.
y1	math.h	double y1(double *x*);	Calculates the Bessel function value of the second kind of order 1.
yn	math.h	double yn(int *n*, double *x*);	Calculates the Bessel function value of the second kind of order *n*.

Note: [1] This function is not supported for files opened with type=record.

Note: [2] This function is not supported for files opened with type=record and mode=ab+, rb+, or wb+.

Note: [3] The ILE C compiler only supports fully buffered and line-buffered streams. Since a block and a line are equal to the record length of the opened file, fully buffered and line-buffered streams are supported in the same way. The `setbuf()` and `setvbuf()` functions have no effect.

Note: [4] This function is not available when LOCALETYPE(*CLD) is specified on the compilation command.

Note: [5] This function is available only when SYSIFCOPT(*IFSIO) is specified on the CRTCMOD or CRTBNDC command.

Note: [6] This function is not available when either LOCALETYPE(*CLD) or SYSIFCOPT(*NOIFSIO) is specified on the compilation command.