Annex B
(informative)

Library summary

B.1 Diagnostics `<assert.h>`

```
NDEBUG
static_assert

void assert(scalar expression);
```

B.2 Complex `<complex.h>`

```
__STDC_NO_COMPLEX__          imaginary
complex                      _Imaginary_I
_Complex_I                   I

#pragma STDC CX_LIMITED_RANGE on-off-switch
double complex cacos(double complex z);
float complex cacosf(float complex z);
long double complex cacosl(long double complex z);
double complex casin(double complex z);
float complex casinf(float complex z);
long double complex casinl(long double complex z);
double complex catan(double complex z);
float complex catanf(float complex z);
long double complex catanl(long double complex z);
double complex ccos(double complex z);
float complex ccosf(float complex z);
long double complex ccosl(long double complex z);
double complex csin(double complex z);
float complex csinf(float complex z);
long double complex csinl(long double complex z);
double complex ctan(double complex z);
float complex ctanf(float complex z);
long double complex ctanl(long double complex z);
double complex cacosh(double complex z);
float complex cacoshf(float complex z);
long double complex cacoshl(long double complex z);
double complex casinh(double complex z);
float complex casinhf(float complex z);
long double complex casinhl(long double complex z);
```

```
double complex catanh(double complex z);
float complex catanhf(float complex z);
long double complex catanhl(long double complex z);
double complex ccosh(double complex z);
float complex ccoshf(float complex z);
long double complex ccoshl(long double complex z);
double complex csinh(double complex z);
float complex csinhf(float complex z);
long double complex csinhl(long double complex z);
double complex ctanh(double complex z);
float complex ctanhf(float complex z);
long double complex ctanhl(long double complex z);
double complex cexp(double complex z);
float complex cexpf(float complex z);
long double complex cexpl(long double complex z);
double complex clog(double complex z);
float complex clogf(float complex z);
long double complex clogl(long double complex z);
double cabs(double complex z);
float cabsf(float complex z);
long double cabsl(long double complex z);
double complex cpow(double complex x, double complex y);
float complex cpowf(float complex x, float complex y);
long double complex cpowl(long double complex x,
     long double complex y);
double complex csqrt(double complex z);
float complex csqrtf(float complex z);
long double complex csqrtl(long double complex z);
double carg(double complex z);
float cargf(float complex z);
long double cargl(long double complex z);
double cimag(double complex z);
float cimagf(float complex z);
long double cimagl(long double complex z);
double complex CMPLX(double x, double y);
float complex CMPLXF(float x, float y);
long double complex CMPLXL(long double x, long double y);
double complex conj(double complex z);
float complex conjf(float complex z);
long double complex conjl(long double complex z);
double complex cproj(double complex z);
```

```
float complex cprojf(float complex z);
long double complex cprojl(long double complex z);
double creal(double complex z);
float crealf(float complex z);
long double creall(long double complex z);
```

B.3 Character handling <ctype.h>

```
int isalnum(int c);
int isalpha(int c);
int isblank(int c);
int iscntrl(int c);
int isdigit(int c);
int isgraph(int c);
int islower(int c);
int isprint(int c);
int ispunct(int c);
int isspace(int c);
int isupper(int c);
int isxdigit(int c);
int tolower(int c);
int toupper(int c);
```

B.4 Errors <errno.h>

```
EDOM          EILSEQ        ERANGE        errno

__STDC_WANT_LIB_EXT1__
errno_t
```

B.5 Floating-point environment <fenv.h>

```
fenv_t              FE_OVERFLOW         FE_TOWARDZERO
fexcept_t           FE_UNDERFLOW        FE_UPWARD
FE_DIVBYZERO        FE_ALL_EXCEPT       FE_DFL_ENV
FE_INEXACT          FE_DOWNWARD
FE_INVALID          FE_TONEAREST
```

```
#pragma STDC FENV_ACCESS on-off-switch
int feclearexcept(int excepts);
int fegetexceptflag(fexcept_t *flagp, int excepts);
int feraiseexcept(int excepts);
int fesetexceptflag(const fexcept_t *flagp,
     int excepts);
int fetestexcept(int excepts);
```

```
int fegetround(void);
int fesetround(int round);
int fegetenv(fenv_t *envp);
int feholdexcept(fenv_t *envp);
int fesetenv(const fenv_t *envp);
int feupdateenv(const fenv_t *envp);
```

B.6 Characteristics of floating types <float.h>

```
FLT_ROUNDS          DBL_DIG             FLT_MAX
FLT_EVAL_METHOD     LDBL_DIG            DBL_MAX
FLT_HAS_SUBNORM     FLT_MIN_EXP         LDBL_MAX
DBL_HAS_SUBNORM     DBL_MIN_EXP         FLT_EPSILON
LDBL_HAS_SUBNORM    LDBL_MIN_EXP        DBL_EPSILON
FLT_RADIX           FLT_MIN_10_EXP      LDBL_EPSILON
FLT_MANT_DIG        DBL_MIN_10_EXP      FLT_MIN
DBL_MANT_DIG        LDBL_MIN_10_EXP     DBL_MIN
LDBL_MANT_DIG       FLT_MAX_EXP         LDBL_MIN
FLT_DECIMAL_DIG     DBL_MAX_EXP         FLT_TRUE_MIN
DBL_DECIMAL_DIG     LDBL_MAX_EXP        DBL_TRUE_MIN
LDBL_DECIMAL_DIG    FLT_MAX_10_EXP      LDBL_TRUE_MIN
DECIMAL_DIG         DBL_MAX_10_EXP
FLT_DIG             LDBL_MAX_10_EXP
```

B.7 Format conversion of integer types <inttypes.h>

```
imaxdiv_t
```

PRId*N*	PRIdLEAST*N*	PRIdFAST*N*	PRIdMAX	PRIdPTR
PRIi*N*	PRIiLEAST*N*	PRIiFAST*N*	PRIiMAX	PRIiPTR
PRIo*N*	PRIoLEAST*N*	PRIoFAST*N*	PRIoMAX	PRIoPTR
PRIu*N*	PRIuLEAST*N*	PRIuFAST*N*	PRIuMAX	PRIuPTR
PRIx*N*	PRIxLEAST*N*	PRIxFAST*N*	PRIxMAX	PRIxPTR
PRIX*N*	PRIXLEAST*N*	PRIXFAST*N*	PRIXMAX	PRIXPTR
SCNd*N*	SCNdLEAST*N*	SCNdFAST*N*	SCNdMAX	SCNdPTR
SCNi*N*	SCNiLEAST*N*	SCNiFAST*N*	SCNiMAX	SCNiPTR
SCNo*N*	SCNoLEAST*N*	SCNoFAST*N*	SCNoMAX	SCNoPTR
SCNu*N*	SCNuLEAST*N*	SCNuFAST*N*	SCNuMAX	SCNuPTR
SCNx*N*	SCNxLEAST*N*	SCNxFAST*N*	SCNxMAX	SCNxPTR

```
intmax_t imaxabs(intmax_t j);
imaxdiv_t imaxdiv(intmax_t numer, intmax_t denom);
intmax_t strtoimax(const char * restrict nptr,
        char ** restrict endptr, int base);
```

```
uintmax_t strtoumax(const char * restrict nptr,
        char ** restrict endptr, int base);
intmax_t wcstoimax(const wchar_t * restrict nptr,
        wchar_t ** restrict endptr, int base);
uintmax_t wcstoumax(const wchar_t * restrict nptr,
        wchar_t ** restrict endptr, int base);
```

B.8 Alternative spellings <iso646.h>

```
and           bitor         not_eq        xor
and_eq        compl         or            xor_eq
bitand        not           or_eq
```

B.9 Sizes of integer types <limits.h>

```
CHAR_BIT      CHAR_MAX      INT_MIN       ULONG_MAX
SCHAR_MIN     MB_LEN_MAX    INT_MAX       LLONG_MIN
SCHAR_MAX     SHRT_MIN      UINT_MAX      LLONG_MAX
UCHAR_MAX     SHRT_MAX      LONG_MIN      ULLONG_MAX
CHAR_MIN      USHRT_MAX     LONG_MAX
```

B.10 Localization <locale.h>

```
struct lconv  LC_ALL        LC_CTYPE      LC_NUMERIC
NULL          LC_COLLATE    LC_MONETARY   LC_TIME
```

```
char *setlocale(int category, const char *locale);
struct lconv *localeconv(void);
```

B.11 Mathematics <math.h>

```
float_t             FP_INFINITE         FP_FAST_FMAL
double_t            FP_NAN              FP_ILOGB0
HUGE_VAL            FP_NORMAL           FP_ILOGBNAN
HUGE_VALF           FP_SUBNORMAL        MATH_ERRNO
HUGE_VALL           FP_ZERO             MATH_ERREXCEPT
INFINITY            FP_FAST_FMA         math_errhandling
NAN                 FP_FAST_FMAF
```

```
#pragma STDC FP_CONTRACT on-off-switch
int fpclassify(real-floating x);
int isfinite(real-floating x);
int isinf(real-floating x);
int isnan(real-floating x);
int isnormal(real-floating x);
int signbit(real-floating x);
```

```
double acos(double x);
float acosf(float x);
long double acosl(long double x);
double asin(double x);
float asinf(float x);
long double asinl(long double x);
double atan(double x);
float atanf(float x);
long double atanl(long double x);
double atan2(double y, double x);
float atan2f(float y, float x);
long double atan2l(long double y, long double x);
double cos(double x);
float cosf(float x);
long double cosl(long double x);
double sin(double x);
float sinf(float x);
long double sinl(long double x);
double tan(double x);
float tanf(float x);
long double tanl(long double x);
double acosh(double x);
float acoshf(float x);
long double acoshl(long double x);
double asinh(double x);
float asinhf(float x);
long double asinhl(long double x);
double atanh(double x);
float atanhf(float x);
long double atanhl(long double x);
double cosh(double x);
float coshf(float x);
long double coshl(long double x);
double sinh(double x);
float sinhf(float x);
long double sinhl(long double x);
double tanh(double x);
float tanhf(float x);
long double tanhl(long double x);
double exp(double x);
float expf(float x);
```

```
long double expl(long double x);
double exp2(double x);
float exp2f(float x);
long double exp2l(long double x);
double expm1(double x);
float expm1f(float x);
long double expm1l(long double x);
double frexp(double value, int *exp);
float frexpf(float value, int *exp);
long double frexpl(long double value, int *exp);
int ilogb(double x);
int ilogbf(float x);
int ilogbl(long double x);
double ldexp(double x, int exp);
float ldexpf(float x, int exp);
long double ldexpl(long double x, int exp);
double log(double x);
float logf(float x);
long double logl(long double x);
double log10(double x);
float log10f(float x);
long double log10l(long double x);
double log1p(double x);
float log1pf(float x);
long double log1pl(long double x);
double log2(double x);
float log2f(float x);
long double log2l(long double x);
double logb(double x);
float logbf(float x);
long double logbl(long double x);
double modf(double value, double *iptr);
float modff(float value, float *iptr);
long double modfl(long double value, long double *iptr);
double scalbn(double x, int n);
float scalbnf(float x, int n);
long double scalbnl(long double x, int n);
double scalbln(double x, long int n);
float scalblnf(float x, long int n);
long double scalblnl(long double x, long int n);
double cbrt(double x);
```

```
float cbrtf(float x);
long double cbrtl(long double x);
double fabs(double x);
float fabsf(float x);
long double fabsl(long double x);
double hypot(double x, double y);
float hypotf(float x, float y);
long double hypotl(long double x, long double y);
double pow(double x, double y);
float powf(float x, float y);
long double powl(long double x, long double y);
double sqrt(double x);
float sqrtf(float x);
long double sqrtl(long double x);
double erf(double x);
float erff(float x);
long double erfl(long double x);
double erfc(double x);
float erfcf(float x);
long double erfcl(long double x);
double lgamma(double x);
float lgammaf(float x);
long double lgammal(long double x);
double tgamma(double x);
float tgammaf(float x);
long double tgammal(long double x);
double ceil(double x);
float ceilf(float x);
long double ceill(long double x);
double floor(double x);
float floorf(float x);
long double floorl(long double x);
double nearbyint(double x);
float nearbyintf(float x);
long double nearbyintl(long double x);
double rint(double x);
float rintf(float x);
long double rintl(long double x);
long int lrint(double x);
long int lrintf(float x);
long int lrintl(long double x);
```

```
long long int llrint(double x);
long long int llrintf(float x);
long long int llrintl(long double x);
double round(double x);
float roundf(float x);
long double roundl(long double x);
long int lround(double x);
long int lroundf(float x);
long int lroundl(long double x);
long long int llround(double x);
long long int llroundf(float x);
long long int llroundl(long double x);
double trunc(double x);
float truncf(float x);
long double truncl(long double x);
double fmod(double x, double y);
float fmodf(float x, float y);
long double fmodl(long double x, long double y);
double remainder(double x, double y);
float remainderf(float x, float y);
long double remainderl(long double x, long double y);
double remquo(double x, double y, int *quo);
float remquof(float x, float y, int *quo);
long double remquol(long double x, long double y,
     int *quo);
double copysign(double x, double y);
float copysignf(float x, float y);
long double copysignl(long double x, long double y);
double nan(const char *tagp);
float nanf(const char *tagp);
long double nanl(const char *tagp);
double nextafter(double x, double y);
float nextafterf(float x, float y);
long double nextafterl(long double x, long double y);
double nexttoward(double x, long double y);
float nexttowardf(float x, long double y);
long double nexttowardl(long double x, long double y);
double fdim(double x, double y);
float fdimf(float x, float y);
long double fdiml(long double x, long double y);
double fmax(double x, double y);
```

```
float fmaxf(float x, float y);
long double fmaxl(long double x, long double y);
double fmin(double x, double y);
float fminf(float x, float y);
long double fminl(long double x, long double y);
double fma(double x, double y, double z);
float fmaf(float x, float y, float z);
long double fmal(long double x, long double y,
     long double z);
int isgreater(real-floating x, real-floating y);
int isgreaterequal(real-floating x, real-floating y);
int isless(real-floating x, real-floating y);
int islessequal(real-floating x, real-floating y);
int islessgreater(real-floating x, real-floating y);
int isunordered(real-floating x, real-floating y);
```

B.12 Nonlocal jumps <setjmp.h>

```
jmp_buf

int setjmp(jmp_buf env);
_Noreturn void longjmp(jmp_buf env, int val);
```

B.13 Signal handling <signal.h>

```
sig_atomic_t  SIG_IGN       SIGILL        SIGTERM
SIG_DFL       SIGABRT       SIGINT
SIG_ERR       SIGFPE        SIGSEGV

void (*signal(int sig, void (*func)(int)))(int);
int raise(int sig);
```

B.14 Alignment `<stdalign.h>`

```
alignas
__alignas_is_defined
```

B.15 Variable arguments `<stdarg.h>`

```
va_list
```

type `va_arg(va_list ap,` *type*`);`
`void va_copy(va_list dest, va_list src);`
`void va_end(va_list ap);`
`void va_start(va_list ap,` *parmN*`);`

B.16 Atomics `<stdatomic.h>`

```
ATOMIC_BOOL_LOCK_FREE     | atomic_uint
ATOMIC_CHAR_LOCK_FREE       atomic_long
ATOMIC_CHAR16_T_LOCK_FREE   atomic_ulong
ATOMIC_CHAR32_T_LOCK_FREE   atomic_llong
ATOMIC_WCHAR_T_LOCK_FREE    atomic_ullong
ATOMIC_SHORT_LOCK_FREE      atomic_char16_t
ATOMIC_INT_LOCK_FREE        atomic_char32_t
ATOMIC_LONG_LOCK_FREE       atomic_wchar_t
ATOMIC_LLONG_LOCK_FREE      atomic_int_least8_t
ATOMIC_POINTER_LOCK_FREE  | atomic_uint_least8_t
ATOMIC_FLAG_INIT            atomic_int_least16_t
memory_order                atomic_uint_least16_t
atomic_flag                 atomic_int_least32_t
memory_order_relaxed      * atomic_uint_least32_t
memory_order_consume        atomic_int_least64_t
memory_order_acquire        atomic_uint_least64_t
memory_order_release        atomic_int_fast8_t
memory_order_acq_rel        atomic_uint_fast8_t
memory_order_seq_cst        atomic_int_fast16_t
atomic_bool               | atomic_uint_fast16_t
atomic_char                 atomic_int_fast32_t
atomic_schar                atomic_uint_fast32_t
atomic_uchar                atomic_int_fast64_t
atomic_short                atomic_uint_fast64_t
atomic_ushort               atomic_intptr_t
atomic_int                  atomic_uintptr_t
```

```
atomic_size_t                atomic_intmax_t
atomic_ptrdiff_t             atomic_uintmax_t

#define ATOMIC_VAR_INIT(C value)
void atomic_init(volatile A *obj, C value);
type kill_dependency(type y);
void atomic_thread_fence(memory_order order);
void atomic_signal_fence(memory_order order);
_Bool atomic_is_lock_free(const volatile A *obj);
void atomic_store(volatile A *object, C desired);
void atomic_store_explicit(volatile A *object,
     C desired, memory_order order);
C atomic_load(volatile A *object);
C atomic_load_explicit(volatile A *object,
     memory_order order);
C atomic_exchange(volatile A *object, C desired);
C atomic_exchange_explicit(volatile A *object,
     C desired, memory_order order);
_Bool atomic_compare_exchange_strong(volatile A *object,
     C *expected, C desired);
_Bool atomic_compare_exchange_strong_explicit(
     volatile A *object, C *expected, C desired,
     memory_order success, memory_order failure);
_Bool atomic_compare_exchange_weak(volatile A *object,
     C *expected, C desired);
_Bool atomic_compare_exchange_weak_explicit(
     volatile A *object, C *expected, C desired,
     memory_order success, memory_order failure);
C atomic_fetch_key(volatile A *object, M operand);
C atomic_fetch_key_explicit(volatile A *object,
     M operand, memory_order order);
_Bool atomic_flag_test_and_set(
     volatile atomic_flag *object);
_Bool atomic_flag_test_and_set_explicit(
     volatile atomic_flag *object, memory_order order);
void atomic_flag_clear(volatile atomic_flag *object);
void atomic_flag_clear_explicit(
     volatile atomic_flag *object, memory_order order);
```

B.17 Boolean type and values `<stdbool.h>`

```
bool
true
false
__bool_true_false_are_defined
```

B.18 Common definitions `<stddef.h>`

```
ptrdiff_t     max_align_t   NULL
size_t        wchar_t
```

`offsetof(`*type*`,` *member-designator*`)`

```
__STDC_WANT_LIB_EXT1__
rsize_t
```

B.19 Integer types `<stdint.h>`

`int`*N*`_t`	`INT_LEAST`*N*`_MIN`	`PTRDIFF_MAX`
`uint`*N*`_t`	`INT_LEAST`*N*`_MAX`	`SIG_ATOMIC_MIN`
`int_least`*N*`_t`	`UINT_LEAST`*N*`_MAX`	`SIG_ATOMIC_MAX`
`uint_least`*N*`_t`	`INT_FAST`*N*`_MIN`	`SIZE_MAX`
`int_fast`*N*`_t`	`INT_FAST`*N*`_MAX`	`WCHAR_MIN`
`uint_fast`*N*`_t`	`UINT_FAST`*N*`_MAX`	`WCHAR_MAX`
`intptr_t`	`INTPTR_MIN`	`WINT_MIN`
`uintptr_t`	`INTPTR_MAX`	`WINT_MAX`
`intmax_t`	`UINTPTR_MAX`	`INT`*N*`_C(value)`
`uintmax_t`	`INTMAX_MIN`	`UINT`*N*`_C(value)`
`INT`*N*`_MIN`	`INTMAX_MAX`	`INTMAX_C(value)`
`INT`*N*`_MAX`	`UINTMAX_MAX`	`UINTMAX_C(value)`
`UINT`*N*`_MAX`	`PTRDIFF_MIN`	

```
__STDC_WANT_LIB_EXT1__
RSIZE_MAX
```

B.20 Input/output <stdio.h>

```
size_t          _IOLBF          FILENAME_MAX  TMP_MAX
FILE            _IONBF          L_tmpnam      stderr
fpos_t          BUFSIZ          SEEK_CUR      stdin
NULL            EOF             SEEK_END      stdout
_IOFBF          FOPEN_MAX       SEEK_SET
int remove(const char *filename);
int rename(const char *old, const char *new);
FILE *tmpfile(void);
char *tmpnam(char *s);
int fclose(FILE *stream);
int fflush(FILE *stream);
FILE *fopen(const char * restrict filename,
     const char * restrict mode);
FILE *freopen(const char * restrict filename,
     const char * restrict mode,
     FILE * restrict stream);
void setbuf(FILE * restrict stream,
     char * restrict buf);
int setvbuf(FILE * restrict stream,
     char * restrict buf,
     int mode, size_t size);
int fprintf(FILE * restrict stream,
     const char * restrict format, ...);
int fscanf(FILE * restrict stream,
     const char * restrict format, ...);
int printf(const char * restrict format, ...);
int scanf(const char * restrict format, ...);
int snprintf(char * restrict s, size_t n,
     const char * restrict format, ...);
int sprintf(char * restrict s,
     const char * restrict format, ...);
int sscanf(const char * restrict s,
     const char * restrict format, ...);
int vfprintf(FILE * restrict stream,
     const char * restrict format, va_list arg);
int vfscanf(FILE * restrict stream,
     const char * restrict format, va_list arg);
int vprintf(const char * restrict format, va_list arg);
int vscanf(const char * restrict format, va_list arg);
```

```
int vsnprintf(char * restrict s, size_t n,
     const char * restrict format, va_list arg);
int vsprintf(char * restrict s,
     const char * restrict format, va_list arg);
int vsscanf(const char * restrict s,
     const char * restrict format, va_list arg);
int fgetc(FILE *stream);
char *fgets(char * restrict s, int n,
     FILE * restrict stream);
int fputc(int c, FILE *stream);
int fputs(const char * restrict s,
     FILE * restrict stream);
int getc(FILE *stream);
int getchar(void);
int putc(int c, FILE *stream);
int putchar(int c);
int puts(const char *s);
int ungetc(int c, FILE *stream);
size_t fread(void * restrict ptr,
     size_t size, size_t nmemb,
     FILE * restrict stream);
size_t fwrite(const void * restrict ptr,
     size_t size, size_t nmemb,
     FILE * restrict stream);
int fgetpos(FILE * restrict stream,
     fpos_t * restrict pos);
int fseek(FILE *stream, long int offset, int whence);
int fsetpos(FILE *stream, const fpos_t *pos);
long int ftell(FILE *stream);
void rewind(FILE *stream);
void clearerr(FILE *stream);
int feof(FILE *stream);
int ferror(FILE *stream);
void perror(const char *s);

__STDC_WANT_LIB_EXT1__
L_tmpnam_s    TMP_MAX_S     errno_t       rsize_t

errno_t tmpfile_s(FILE * restrict * restrict streamptr);
errno_t tmpnam_s(char *s, rsize_t maxsize);
```

```
errno_t fopen_s(FILE * restrict * restrict streamptr,
     const char * restrict filename,
     const char * restrict mode);
errno_t freopen_s(FILE * restrict * restrict newstreamptr,
     const char * restrict filename,
     const char * restrict mode,
     FILE * restrict stream);
int fprintf_s(FILE * restrict stream,
     const char * restrict format, ...);
int fscanf_s(FILE * restrict stream,
     const char * restrict format, ...);
int printf_s(const char * restrict format, ...);
int scanf_s(const char * restrict format, ...);
int snprintf_s(char * restrict s, rsize_t n,
     const char * restrict format, ...);
int sprintf_s(char * restrict s, rsize_t n,
     const char * restrict format, ...);
int sscanf_s(const char * restrict s,
     const char * restrict format, ...);
int vfprintf_s(FILE * restrict stream,
     const char * restrict format,
     va_list arg);
int vfscanf_s(FILE * restrict stream,
     const char * restrict format,
     va_list arg);
int vprintf_s(const char * restrict format,
     va_list arg);
int vscanf_s(const char * restrict format,
     va_list arg);
int vsnprintf_s(char * restrict s, rsize_t n,
     const char * restrict format,
     va_list arg);
int vsprintf_s(char * restrict s, rsize_t n,
     const char * restrict format,
     va_list arg);
int vsscanf_s(const char * restrict s,
     const char * restrict format,
     va_list arg);
char *gets_s(char *s, rsize_t n);
```

B.21 General utilities `<stdlib.h>`

```
size_t        ldiv_t        EXIT_FAILURE  MB_CUR_MAX
wchar_t       lldiv_t       EXIT_SUCCESS
div_t         NULL          RAND_MAX

double atof(const char *nptr);
int atoi(const char *nptr);
long int atol(const char *nptr);
long long int atoll(const char *nptr);
double strtod(const char * restrict nptr,
     char ** restrict endptr);
float strtof(const char * restrict nptr,
     char ** restrict endptr);
long double strtold(const char * restrict nptr,
     char ** restrict endptr);
long int strtol(const char * restrict nptr,
     char ** restrict endptr, int base);
long long int strtoll(const char * restrict nptr,
     char ** restrict endptr, int base);
unsigned long int strtoul(
     const char * restrict nptr,
     char ** restrict endptr, int base);
unsigned long long int strtoull(
     const char * restrict nptr,
     char ** restrict endptr, int base);
int rand(void);
void srand(unsigned int seed);
void *aligned_alloc(size_t alignment, size_t size);
void *calloc(size_t nmemb, size_t size);
void free(void *ptr);
void *malloc(size_t size);
void *realloc(void *ptr, size_t size);
_Noreturn void abort(void);
int atexit(void (*func)(void));
int at_quick_exit(void (*func)(void));
_Noreturn void exit(int status);
_Noreturn void _Exit(int status);
char *getenv(const char *name);
_Noreturn void quick_exit(int status);
int system(const char *string);
```

```
void *bsearch(const void *key, const void *base,
     size_t nmemb, size_t size,
     int (*compar)(const void *, const void *));
void qsort(void *base, size_t nmemb, size_t size,
     int (*compar)(const void *, const void *));
int abs(int j);
long int labs(long int j);
long long int llabs(long long int j);
div_t div(int numer, int denom);
ldiv_t ldiv(long int numer, long int denom);
lldiv_t lldiv(long long int numer,
     long long int denom);
int mblen(const char *s, size_t n);
int mbtowc(wchar_t * restrict pwc,
     const char * restrict s, size_t n);
int wctomb(char *s, wchar_t wchar);
size_t mbstowcs(wchar_t * restrict pwcs,
     const char * restrict s, size_t n);
size_t wcstombs(char * restrict s,
     const wchar_t * restrict pwcs, size_t n);

__STDC_WANT_LIB_EXT1__
errno_t
rsize_t
constraint_handler_t

constraint_handler_t set_constraint_handler_s(
     constraint_handler_t handler);
void abort_handler_s(
     const char * restrict msg,
     void * restrict ptr,
     errno_t error);
void ignore_handler_s(
     const char * restrict msg,
     void * restrict ptr,
     errno_t error);
errno_t getenv_s(size_t * restrict len,
          char * restrict value, rsize_t maxsize,
          const char * restrict name);
```

```
void *bsearch_s(const void *key, const void *base,
     rsize_t nmemb, rsize_t size,
     int (*compar)(const void *k, const void *y,
                   void *context),
     void *context);
errno_t qsort_s(void *base, rsize_t nmemb, rsize_t size,
     int (*compar)(const void *x, const void *y,
                   void *context),
     void *context);
errno_t wctomb_s(int * restrict status,
     char * restrict s,
     rsize_t smax,
     wchar_t wc);
errno_t mbstowcs_s(size_t * restrict retval,
     wchar_t * restrict dst, rsize_t dstmax,
     const char * restrict src, rsize_t len);
errno_t wcstombs_s(size_t * restrict retval,
     char * restrict dst, rsize_t dstmax,
     const wchar_t * restrict src, rsize_t len);
```

B.22 _Noreturn <stdnoreturn.h>

```
noreturn
```

B.23 String handling <string.h>

```
size_t
NULL
```

```
void *memcpy(void * restrict s1,
     const void * restrict s2, size_t n);
void *memmove(void *s1, const void *s2, size_t n);
char *strcpy(char * restrict s1,
     const char * restrict s2);
char *strncpy(char * restrict s1,
     const char * restrict s2, size_t n);
char *strcat(char * restrict s1,
     const char * restrict s2);
char *strncat(char * restrict s1,
     const char * restrict s2, size_t n);
int memcmp(const void *s1, const void *s2, size_t n);
int strcmp(const char *s1, const char *s2);
int strcoll(const char *s1, const char *s2);
int strncmp(const char *s1, const char *s2, size_t n);
```

```
size_t strxfrm(char * restrict s1,
     const char * restrict s2, size_t n);
void *memchr(const void *s, int c, size_t n);
char *strchr(const char *s, int c);
size_t strcspn(const char *s1, const char *s2);
char *strpbrk(const char *s1, const char *s2);
char *strrchr(const char *s, int c);
size_t strspn(const char *s1, const char *s2);
char *strstr(const char *s1, const char *s2);
char *strtok(char * restrict s1,
     const char * restrict s2);
void *memset(void *s, int c, size_t n);
char *strerror(int errnum);
size_t strlen(const char *s);

__STDC_WANT_LIB_EXT1__
errno_t
rsize_t

errno_t memcpy_s(void * restrict s1, rsize_t s1max,
     const void * restrict s2, rsize_t n);
errno_t memmove_s(void *s1, rsize_t s1max,
     const void *s2, rsize_t n);
errno_t strcpy_s(char * restrict s1,
     rsize_t s1max,
     const char * restrict s2);
errno_t strncpy_s(char * restrict s1,
     rsize_t s1max,
     const char * restrict s2,
     rsize_t n);
errno_t strcat_s(char * restrict s1,
     rsize_t s1max,
     const char * restrict s2);
errno_t strncat_s(char * restrict s1,
     rsize_t s1max,
     const char * restrict s2,
     rsize_t n);
char *strtok_s(char * restrict s1,
     rsize_t * restrict s1max,
     const char * restrict s2,
     char ** restrict ptr);
errno_t memset_s(void *s, rsize_t smax, int c, rsize_t n)
```

```
errno_t strerror_s(char *s, rsize_t maxsize,
     errno_t errnum);
size_t strerrorlen_s(errno_t errnum);
size_t strnlen_s(const char *s, size_t maxsize);
```

B.24 Type-generic math `<tgmath.h>`

```
acos          sqrt          fmod          nextafter
asin          fabs          frexp         nexttoward
atan          atan2         hypot         remainder
acosh         cbrt          ilogb         remquo
asinh         ceil          ldexp         rint
atanh         copysign      lgamma        round
cos           erf           llrint        scalbn
sin           erfc          llround       scalbln
tan           exp2          log10         tgamma
cosh          expm1         log1p         trunc
sinh          fdim          log2          carg
tanh          floor         logb          cimag
exp           fma           lrint         conj
log           fmax          lround        cproj
pow           fmin          nearbyint     creal
```

B.25 Threads `<threads.h>`

```
thread_local                | once_flag
ONCE_FLAG_INIT                mtx_plain                  *
TSS_DTOR_ITERATIONS           mtx_recursive
cnd_t                         mtx_timed
thrd_t                        thrd_timedout              |
tss_t                         thrd_success
mtx_t                         thrd_busy
tss_dtor_t                    thrd_error
thrd_start_t                  thrd_nomem
```

```
void call_once(once_flag *flag, void (*func)(void));
int cnd_broadcast(cnd_t *cond);
void cnd_destroy(cnd_t *cond);
int cnd_init(cnd_t *cond);
int cnd_signal(cnd_t *cond);
int cnd_timedwait(cnd_t *restrict cond,                  |
     mtx_t *restrict mtx,                                |
     const struct timespec *restrict ts);                |
int cnd_wait(cnd_t *cond, mtx_t *mtx);
```

```
void mtx_destroy(mtx_t *mtx);
int mtx_init(mtx_t *mtx, int type);
int mtx_lock(mtx_t *mtx);
int mtx_timedlock(mtx_t *restrict mtx,
     const struct timespec *restrict ts);
int mtx_trylock(mtx_t *mtx);
int mtx_unlock(mtx_t *mtx);
int thrd_create(thrd_t *thr, thrd_start_t func,
     void *arg);
thrd_t thrd_current(void);
int thrd_detach(thrd_t thr);
int thrd_equal(thrd_t thr0, thrd_t thr1);
_Noreturn void thrd_exit(int res);
int thrd_join(thrd_t thr, int *res);
int thrd_sleep(const struct timespec *duration,
     struct timespec *remaining);
void thrd_yield(void);
int tss_create(tss_t *key, tss_dtor_t dtor);
void tss_delete(tss_t key);
void *tss_get(tss_t key);
int tss_set(tss_t key, void *val);
```

B.26 Date and time <time.h>

```
NULL              size_t            struct timespec
CLOCKS_PER_SEC    clock_t           struct tm
TIME_UTC          time_t
```

```
clock_t clock(void);
double difftime(time_t time1, time_t time0);
time_t mktime(struct tm *timeptr);
time_t time(time_t *timer);
int timespec_get(timespec *ts, int base);
char *asctime(const struct tm *timeptr);
char *ctime(const time_t *timer);
struct tm *gmtime(const time_t *timer);
struct tm *localtime(const time_t *timer);
size_t strftime(char * restrict s,
     size_t maxsize,
     const char * restrict format,
     const struct tm * restrict timeptr);
```

```
__STDC_WANT_LIB_EXT1__
```

```
errno_t
rsize_t

errno_t asctime_s(char *s, rsize_t maxsize,
     const struct tm *timeptr);
errno_t ctime_s(char *s, rsize_t maxsize,
     const time_t *timer);
struct tm *gmtime_s(const time_t * restrict timer,
     struct tm * restrict result);
struct tm *localtime_s(const time_t * restrict timer,
     struct tm * restrict result);
```

B.27 Unicode utilities <uchar.h>

```
mbstate_t     size_t        char16_t      char32_t

size_t mbrtoc16(char16_t * restrict pc16,
     const char * restrict s, size_t n,
     mbstate_t * restrict ps);
size_t c16rtomb(char * restrict s, char16_t c16,
     mbstate_t * restrict ps);
size_t mbrtoc32(char32_t * restrict pc32,
     const char * restrict s, size_t n,
     mbstate_t * restrict ps);
size_t c32rtomb(char * restrict s, char32_t c32,
     mbstate_t * restrict ps);
```

B.28 Extended multibyte/wide character utilities <wchar.h>

```
wchar_t           wint_t            WCHAR_MAX
size_t            struct tm         WCHAR_MIN
mbstate_t         NULL              WEOF

int fwprintf(FILE * restrict stream,
     const wchar_t * restrict format, ...);
int fwscanf(FILE * restrict stream,
     const wchar_t * restrict format, ...);
int swprintf(wchar_t * restrict s, size_t n,
     const wchar_t * restrict format, ...);
int swscanf(const wchar_t * restrict s,
     const wchar_t * restrict format, ...);
int vfwprintf(FILE * restrict stream,
     const wchar_t * restrict format, va_list arg);
```

```
int vfwscanf(FILE * restrict stream,
     const wchar_t * restrict format, va_list arg);
int vswprintf(wchar_t * restrict s, size_t n,
     const wchar_t * restrict format, va_list arg);
int vswscanf(const wchar_t * restrict s,
     const wchar_t * restrict format, va_list arg);
int vwprintf(const wchar_t * restrict format,
     va_list arg);
int vwscanf(const wchar_t * restrict format,
     va_list arg);
int wprintf(const wchar_t * restrict format, ...);
int wscanf(const wchar_t * restrict format, ...);
wint_t fgetwc(FILE *stream);
wchar_t *fgetws(wchar_t * restrict s, int n,
     FILE * restrict stream);
wint_t fputwc(wchar_t c, FILE *stream);
int fputws(const wchar_t * restrict s,
     FILE * restrict stream);
int fwide(FILE *stream, int mode);
wint_t getwc(FILE *stream);
wint_t getwchar(void);
wint_t putwc(wchar_t c, FILE *stream);
wint_t putwchar(wchar_t c);
wint_t ungetwc(wint_t c, FILE *stream);
double wcstod(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr);
float wcstof(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr);
long double wcstold(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr);
long int wcstol(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr, int base);
long long int wcstoll(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr, int base);
unsigned long int wcstoul(const wchar_t * restrict nptr,
     wchar_t ** restrict endptr, int base);
unsigned long long int wcstoull(
     const wchar_t * restrict nptr,
     wchar_t ** restrict endptr, int base);
```

```
wchar_t *wcscpy(wchar_t * restrict s1,
     const wchar_t * restrict s2);
wchar_t *wcsncpy(wchar_t * restrict s1,
     const wchar_t * restrict s2, size_t n);
wchar_t *wmemcpy(wchar_t * restrict s1,
     const wchar_t * restrict s2, size_t n);
wchar_t *wmemmove(wchar_t *s1, const wchar_t *s2,
     size_t n);
wchar_t *wcscat(wchar_t * restrict s1,
     const wchar_t * restrict s2);
wchar_t *wcsncat(wchar_t * restrict s1,
     const wchar_t * restrict s2, size_t n);
int wcscmp(const wchar_t *s1, const wchar_t *s2);
int wcscoll(const wchar_t *s1, const wchar_t *s2);
int wcsncmp(const wchar_t *s1, const wchar_t *s2,
     size_t n);
size_t wcsxfrm(wchar_t * restrict s1,
     const wchar_t * restrict s2, size_t n);
int wmemcmp(const wchar_t *s1, const wchar_t *s2,
     size_t n);
wchar_t *wcschr(const wchar_t *s, wchar_t c);
size_t wcscspn(const wchar_t *s1, const wchar_t *s2);
wchar_t *wcspbrk(const wchar_t *s1, const wchar_t *s2);
wchar_t *wcsrchr(const wchar_t *s, wchar_t c);
size_t wcsspn(const wchar_t *s1, const wchar_t *s2);
wchar_t *wcsstr(const wchar_t *s1, const wchar_t *s2);
wchar_t *wcstok(wchar_t * restrict s1,
     const wchar_t * restrict s2,
     wchar_t ** restrict ptr);
wchar_t *wmemchr(const wchar_t *s, wchar_t c, size_t n);
size_t wcslen(const wchar_t *s);
wchar_t *wmemset(wchar_t *s, wchar_t c, size_t n);
size_t wcsftime(wchar_t * restrict s, size_t maxsize,
     const wchar_t * restrict format,
     const struct tm * restrict timeptr);
wint_t btowc(int c);
int wctob(wint_t c);
int mbsinit(const mbstate_t *ps);
size_t mbrlen(const char * restrict s, size_t n,
     mbstate_t * restrict ps);
```

```
size_t mbrtowc(wchar_t * restrict pwc,
     const char * restrict s, size_t n,
     mbstate_t * restrict ps);
size_t wcrtomb(char * restrict s, wchar_t wc,
     mbstate_t * restrict ps);
size_t mbsrtowcs(wchar_t * restrict dst,
     const char ** restrict src, size_t len,
     mbstate_t * restrict ps);
size_t wcsrtombs(char * restrict dst,
     const wchar_t ** restrict src, size_t len,
     mbstate_t * restrict ps);
```

```
__STDC_WANT_LIB_EXT1__
errno_t
rsize_t
```

```
int fwprintf_s(FILE * restrict stream,
     const wchar_t * restrict format, ...);
int fwscanf_s(FILE * restrict stream,
     const wchar_t * restrict format, ...);
int snwprintf_s(wchar_t * restrict s,
     rsize_t n,
     const wchar_t * restrict format, ...);
int swprintf_s(wchar_t * restrict s, rsize_t n,
     const wchar_t * restrict format, ...);
int swscanf_s(const wchar_t * restrict s,
     const wchar_t * restrict format, ...);
int vfwprintf_s(FILE * restrict stream,
     const wchar_t * restrict format,
     va_list arg);
int vfwscanf_s(FILE * restrict stream,
     const wchar_t * restrict format, va_list arg);
int vsnwprintf_s(wchar_t * restrict s,
     rsize_t n,
     const wchar_t * restrict format,
     va_list arg);
int vswprintf_s(wchar_t * restrict s,
     rsize_t n,
     const wchar_t * restrict format,
     va_list arg);
```

```
int vswscanf_s(const wchar_t * restrict s,
     const wchar_t * restrict format,
     va_list arg);
int vwprintf_s(const wchar_t * restrict format,
     va_list arg);
int vwscanf_s(const wchar_t * restrict format,
     va_list arg);
int wprintf_s(const wchar_t * restrict format, ...);
int wscanf_s(const wchar_t * restrict format, ...);
errno_t wcscpy_s(wchar_t * restrict s1,
     rsize_t s1max,
     const wchar_t * restrict s2);
errno_t wcsncpy_s(wchar_t * restrict s1,
     rsize_t s1max,
     const wchar_t * restrict s2,
     rsize_t n);
errno_t wmemcpy_s(wchar_t * restrict s1,
     rsize_t s1max,
     const wchar_t * restrict s2,
     rsize_t n);
errno_t wmemmove_s(wchar_t *s1, rsize_t s1max,
     const wchar_t *s2, rsize_t n);
errno_t wcscat_s(wchar_t * restrict s1,
     rsize_t s1max,
     const wchar_t * restrict s2);
errno_t wcsncat_s(wchar_t * restrict s1,
     rsize_t s1max,
     const wchar_t * restrict s2,
     rsize_t n);
wchar_t *wcstok_s(wchar_t * restrict s1,
     rsize_t * restrict s1max,
     const wchar_t * restrict s2,
     wchar_t ** restrict ptr);
size_t wcsnlen_s(const wchar_t *s, size_t maxsize);
errno_t wcrtomb_s(size_t * restrict retval,
     char * restrict s, rsize_t smax,
     wchar_t wc, mbstate_t * restrict ps);
```

```
errno_t mbsrtowcs_s(size_t * restrict retval,
     wchar_t * restrict dst, rsize_t dstmax,
     const char ** restrict src, rsize_t len,
     mbstate_t * restrict ps);
errno_t wcsrtombs_s(size_t * restrict retval,
     char * restrict dst, rsize_t dstmax,
     const wchar_t ** restrict src, rsize_t len,
     mbstate_t * restrict ps);
```

B.29 Wide character classification and mapping utilities <wctype.h>

```
wint_t        wctrans_t     wctype_t      WEOF
```

```
int iswalnum(wint_t wc);
int iswalpha(wint_t wc);
int iswblank(wint_t wc);
int iswcntrl(wint_t wc);
int iswdigit(wint_t wc);
int iswgraph(wint_t wc);
int iswlower(wint_t wc);
int iswprint(wint_t wc);
int iswpunct(wint_t wc);
int iswspace(wint_t wc);
int iswupper(wint_t wc);
int iswxdigit(wint_t wc);
int iswctype(wint_t wc, wctype_t desc);
wctype_t wctype(const char *property);
wint_t towlower(wint_t wc);
wint_t towupper(wint_t wc);
wint_t towctrans(wint_t wc, wctrans_t desc);
wctrans_t wctrans(const char *property);
```