Math, Time, and Other Library Functions

- `#include <cmath>`

Math

Trig

- `sin/cos/tan`
- `axxx`: inverse of `xxx`
- `xxxh`: hyperbolic `xxx`

Other

- `abs(x)`: absolute value
- `ceil(x)`: ceiling
- `exp(x)`: exponent
- `floor(x)`: floor
- `log(x)`: natural logarithm
- `log10(x)`: logarithm in base 10
- `pow(base, exponent)`: base raised to exp
- `sqrt(x)`: square root

C Data and Time

Common use cases

1. `time` function to get current time `time(NULL)` gives current time.
2. `localtime(&tm)` breaks `tm` into components
3. `strftime` display time in string format
4. `ctime` display standard timestamp

- `asctime(tm_ptr)`: pointer to a `tm` struct, returns a C-string format
- `clock()`: returns the number of internal clock ticks (~1/1000 secs) since the program bega
- `ctime(time_t_ptr)`: C-string timestamp
- `difftime(time_t_1, time_t_2)`: C-string timestamp
- `gmtime(time_t_ptr)`: produces GMT time by adjusting system timezone
- `localtime(time_t_ptr)`: returns a `tm` struct
- `mktime(tm_ptr)`: returns `time_t` val corresponding to the `tm` struct
- `strftime(dest_str, n, fmt_str, tm_str)`: writes formatted date/time to *dest_str*
- `time(time_t_ptr)`: returns the current time as a `time_t` value

`TM` data structure

`tm` struct contains the time/date information broken down into components.

Members

- `tm_sec`: seconds 0-59
- `tm_min`: minutes 0-59
- `tm_hour`: hours 0-59
- `tm_mday`: day of month 1-31
- `tm_mon`: day of month 0-11
- `tm_year`: number of years since 1900
- `tm_wday`: day of week ranging from 0-6
- `tm_yday`: day of year ranging from 0-365
- `tm_isdst`: indicates whether daylight saving time is in effect

Date/Time format Specifiers

- `%a`: day of the week, three letter abrv
- `%A`: day of the week, full
- `%b`: name of the month, three letter abrv
- `%B`: name of the month, full
- `%c`: complete data/time *mm/dd/yy hh:mm:ss*
- `%d`: day of moths as 2 digit number
- `%H`: hour as 2 digit number 00-23
- `%I`: hour as 2 digit number 00-11
- `%j`: day of year, 3 digit number
- `%m`: month as 2 digit number
- `%M`: minutes as 2 digit number
- `%p`: two char AM or PM
- `%S`: second as 2 digit number
- `%U`: week as two digit number 0-53
- `%w`: weekday as decimal
- `%W`: week as number, same a `%U` but first day is Monday
- `%x`: locale-dependent date representation
- `%X`: locale-dependent time representation
- `%y`: string with last 2 digits of year
- `%Y`: 4 digit string of year
- `%Z`: time zone

String-to-Number conversions

- `atof(s)`: digit string to double
- `atoi(s)`: digit string to int
- `atol(s)`: digit string to long
- `strtod(s, ptr_to_s)`: sets *ptr_to_s* to first char not successfully read, useful for strings with multiple digits

Memory Allocation Functions

In C++ `new` and `delete` are preferable. The advantage of the C functions is that they include `realloc` which resizes an array while preserving the contents.

- `calloc(size, count)`: attempts to allocate a memory block *size x count*. If the function succeeds `*void` returned.
- `free(ptr)`: releases memory previously allocated by `calloc` / `malloc`
- `malloc(size)`: attempts to allocate memory of block *size* large
- `realloc(ptr, new_size)`: reallocates the memory of block pointed to by *ptr* adjusting for memory block to have new size.

Standard C randomisation functions

- `rand()`: returns a pseudo-random number
- `srand()`: sets the seed for the random-number generator

Searching and Sort

- `bsearch`: searches an array for a target value, returns a pointer if found or null pointer if not
- `qsort`: sorts all the values of an array, leaving it in ascending order - uses the quick sort algorithm

Misc

- `abort()`: terminates the program
- `atexit(funct)`: registers an exit function called on termination
- `exit(n_status)`: causes the program to end normally returning status code *n_status*
- `getenv(env_var_str)`: returns the specified environment variable as a C-string
- `ldiv(n1, n2)`: division but returns as long integer
- `system(command_str)`: send *command_str* to the system