1 Timing Functions

In order to compute the processor time, the difference between values returned by two different calls to clock(), one at the start and other at the end of the program is used. To convert the value to seconds, it needs to be divided by a macro $CLOCKS\_PER\_SEC$.
The **clock()** time may advance faster or slower than the actual wall clock. It depends on how the operating system allocates the resources for the process.
If the processor is shared by other processes, the **clock()** time may advance slower than the wall clock. While if the current process is executed in a multi-threaded system, the **clock()** time may advance faster than wall clock.
clock() prototype: clock_t **clock()**;
clock() return value:

- On success, the clock() function returns the processor time used by the program till now.
- On failure, it returns -1 that is casted to the type clock_t.

Using the function int clock_gettime(clockid_t clock_id, struct timespec *tp), we can pass either one of the two following IDS: CLOCK_PROCESS_CPUTIME_ID or CLOCK_THREAD_CPUTIME_ID.
The former is a high-resolution per-process timer from the CPU and the latter is thread-specific CPU-time clock.