Useful links and points to get started:

- Prepare the environment for developing Linux kernel with qemu
- You can use the following website to browse through the Linux source:
  https://elixir.bootlin.com/linux/v6.0.19/source
- **A tip on browsing implementations of already existing syscalls:** Sometimes you may want to browse the implementation of an already existing system call and syscall implementations can be tough to find on bootlin. This happens because of the way system calls are defined in a standard way where macros are used which prepend prefixes to syscall names. Also, macro parameters are often not indexed and hence you wouldn't find a reference or a definition.

  Syscalls are defined using macros like SYSCALL_DEFINEX(), where X is the number of arguments the syscall accepts. You can use these macros to track your system call. Tracking the use of these macros will give you a list of files these are referenced on and then you can use common sense to figure out which file is likely to have the system call you are looking for.

  You can also use any code editor to perform a text or pattern search but given the size of the codebase, it may take a lot of time and browsing.
- A brief guide to priority and nice values in the linux ecosystem
- Few tips for a faster compilation:
  - Every time you make some changes to the code, **DO NOT use**

    ```
    make menuconfig
    ```

    or else it will trigger the complete compilation every time. You only need to make config once at the start.
  - You may want to compile it only for your architecture, example for x86_64 you can use

    ```
    make ARCH=x86_64
    ```

  - Use make parallel build with -j option as follows:

    ```
    make -jN
    ```

    Where N can be the number of cpus in your system, to get the number of core for your system, you can use the following command:

    ```
    lscpu | grep "CPU(s):"
    ```

  - You may also want to create a minimal configuration for your build using,

    ```
    make tinyconfig
    ```