3.1

Where in system/ is the source code of nulluser() defined?

nulluser() is defined in initialize.c

Where in the XINU source code is this ancestor of all processes made up/named and what name (i.e., PID) does it give itself?

Ancestor process is called “prnul” and given the pid 0.

Does nulluser() ever return after being called by the code in start.S?

nulluser() contains an infinite loop at the end do it doesn’t return.

What does the ancestor process running the function nulluser() do for the rest of its existence?

Runs infinitely to keep processor occupied when no other process is executing.

What function is called after nulluser() in start.S?

nulluser() shouldn’t return so no other function should be called after that. But if nulluser() does return **halt()** will be called.

What happens if you remove this function call from start.S?

The function is similar to an infinite loop.

```
halt:
        jmp halt
        ret
```

It keeps the OS in the process table throughout the session. If removed from the start.S code and nulluser() returns due to an error, the OS code will finish execution. XINU will not run as before since the OS will exit/return after the initialization has taken place.

3.2

In XINU what happens after a new process is created using create()?

The process table entry is created, the stack is initialized, return addresses are set in the stack for the function to return, instruction pointer is set to the new function & the new process is set to “suspend state”.

What happens in Linux after a new process is created using fork()?

The CPU scheduler decides which process runs first since fork() initializes the conditions necessary for the new process and adds the process to the scheduling queue.

Who runs first: parent or child?

Xinu: Parent runs first.
Linux: Depends on the scheduler. (Using the code in this post, there is no predictable order of execution)

How does the same app programmer write/port the C code so that it runs over XINU?

XINU as an OS is run from memory so it is not possible to run code from a file in disk. The OS image has to be recompiled every time the programmer wants to add a new app to the OS. That is, the programmer will have to modify the source code for XINU, recompile the OS image and execute the command through a shell command in XINU.

What are the disadvantages of writing app code under XINU? What are the advantages?

Disadvantage: Small changes require the OS to be recompiled every time and this increases development time. This also means large apps increase the size of the OS image and reduce the free space available on the memory even when the app is not being used frequently.
Advantage: The XINU architecture allows the whole OS + apps to fit on memory and this is useful for small devices that don't need a dedicated disk to store other programs.

3.3

Customize the welcome message printed by XINU so that it prominently displays your name and user ID

In turnin folder.

3.4

In turnin. (forever0() added to main() & Process table entry printed out before and after).

What ultimately happens to XINU when no processes remain. Explain what you find.

The OS prints the message "All user processes have completed." The xdone() method is called and machine executes halt().

If XINU were to be run on a mobile platform such as a smartphone, what change would you make in the software based shutdown procedure described above and why?

On a smartphone, the power is limited by a battery so the shutdown procedure should return from halt() instead of executing an infinite loop to save battery power instead of running the CPU for no reason.

Bonus Problem

Added a new command, pcount, that prints the total number of processes in the system

Usage:

```
xsh $ pcount
```

```
No. of active processes: 5
```

Added the command code in file xsh_pcount.c in shell/ folder.