【C】Day8

Course	Advanced C
Study Date	@April 14, 2022

【Ch8】The UNIX Interface

The Unix operating system provides its services through a set of system calls, which are in effect functions within the operating system that might be called by user programs.

Ch7 is concerned with an input/output interface that is uniform across operating systems.

8.1 File Descriptors

In the Unix OS, all input and output is done by reading or writing files because all devices are files in the file system.

Before we read or write a file, we must inform the system of our intent to do so, a process called opening the file. The system checks our right to do so. (Does the file exist? Do we have permission to access it?)

If all if well, open returns to the program a small non-negative integer called a file descriptor.

When the command interpreter runs a program, three files are open, with file descriptors 0, 1, and 2, called the standard input, the standard output, and the standard error.

The user of a program can redirect I/O to and from files with `<` and `>`:

```
prog <infile >outfile
```

In this case, the shell changes the default assignments for file descriptors 0 and 1 to the named files.

Normally file descriptor 2 remains attached to the screen, so error messages can go there.

8.2 Low Level I/O-Read and Write

Input and output uses the `read` and `write` system calls, which are accessed from C programs through two functions called `read` and `write`.

- For both, the first argument is a file descriptor.
- The second argument is a character array in our program where the data is to go to or come from.
- The third argument is the number of bytes to be transferred.

```
int n_read = read(int fd, char *buf, int n);
int n_write = read(int fd, char *buf, int n);
```

Each call returns a count of the number of bytes transferred.

A return value of zero bytes implies end of file, and -1 indicates an error of some sort.