【C】Day8(2)

Course	Advanced C
Study Date	@April 14, 2022

【Ch8】The Unix System Interface

8.3 Open, Creat, Close, Unlink

We must explicitly open files(other than stdin, stdout, and stderrr) in order to read or write them.

There are two system calls for this, `open` and `creat`.

`open` returns -1 if any error occurs.

```
#include <fcntl.h>

int fd;
int open(char *name, int flags, int perms);

fd = open(name, flags, perms);
```

- The `name` argument is a character string containing the filename.
- The `flags` argument is an `int` that specifies how the file is to be opened; the main values are:

```
O_RDONLY // Open for reading only
O_WRONLY // Open for writing only
O_RDWR // Open for both reading and writing
O_CREAT // Create the file if not already existed
O_TRUNC // Truncate the file length to 0 at next wrtie
O_APPEND // Write to the end of the file
```

These constants are defined in `<fcntl.h>` on Unix systems.

To open an existing file for reading:

```
int fd = open(name, O_RDONLY, 0);
```

- The `perms` argument is used to specify the permission when creating a new file. It specifies the permission with one bit that specifies if a file is a directory or a file and nine bits that specify the permission of reads, writes, and executes by the owner, users of the group, and the world.

  ```
  0 111 110 110 // A normal file that allows reads, writes, and executes by the owner
  // Permission for reads, writes for users of the group and the world.
  ```

For example, 0755 specifies read, write, and execute permission for the owner, and read and execute permission for the group and the world user.

It is an error to try to open a file that does not exist. The system call `creat` is provided to create new files, or to re-write old ones:

```
int creat(char *name, int perms);
fd = creat(name, perms);
```

The function `unlink(char *name`) removes the file name from the file system.