任务要求

OOM 模式：
应用会试图打开 `/proc/sys/vm/overcommit_memory` 文件并读取其中的信息。这个文件包含了“内核如何处理内存溢出情况”的信息。它比较复杂且不必要，我们不用实现，读取时直接默认返回一个 `"0"` 即可（注意不是返回空，是一个 ASCII 字符 `0`）

修改代码

crates/linux_syscall_api/src/syscall_fs/imp/io.rs

```
pub fn syscall_openat(args: [usize; 6]) -> SyscallResult {
    let fd = args[0];
    let path = args[1] as *const u8;
    let flags = args[2];
    let _mode = args[3] as u8;
    let force_dir = OpenFlags::from(flags).is_dir();
    let path = solve_path(fd, Some(path), force_dir)?;
    if path.path() == "/proc/sys/vm/overcommit_memory" {
        return Ok('0' as isize);
    }
```

根据commit修改后

```
// TODO: Implement the real content of overcommit_memory
fn oominfo() -> &'static str {
    "0"
}
```

```
let mem_file = axfs::api::lookup("proc/meminfo").unwrap();
mem_file.write_at(0, meminfo().as_bytes()).unwrap();
let oom_file = axfs::api::lookup("/proc/sys/vm/overcommit_memory").unwrap();
oom_file.write_at(0, oominfo().as_bytes()).unwrap();
```

```
root@ubuntu:/home/josen/dev/Starry-bqs/crates/axstarry# git diff
diff --git a/src/file.rs b/src/file.rs
index 14b95ca..e8573b6 100644
--- a/src/file.rs
+++ b/src/file.rs
@@ -61,6 +61,11 @@ DirectMap2M:    23283712 kB
 DirectMap1G:     3145728 kB"
 }

+/// TODO: Implement the real content of overcommit_memory
+fn oominfo() -> &'static str {
+    '0'
+}
+
 /// 在执行系统调用前初始化文件系统
 ///
 /// 包括建立软连接，提前准备好一系列的文件与文件夹
@@ -164,8 +169,10 @@ pub fn fs_init() {
         );
     }

-    let file = axfs::api::lookup("proc/meminfo").unwrap();
-    file.write_at(0, meminfo().as_bytes()).unwrap();
+    let mem_file = axfs::api::lookup("proc/meminfo").unwrap();
+    mem_file.write_at(0, meminfo().as_bytes()).unwrap();
+    let oom_file = axfs::api::lookup("/proc/sys/vm/overcommit_memory").unwrap();
+    oom_file.write_at(0, oominfo().as_bytes()).unwrap();
     // create the file for the lmbench testcase
     let _ = new_file("/lat_sig", &(FileFlags::CREATE | FileFlags::RDWR));

root@ubuntu:/home/josen/dev/Starry-bqs/crates/axstarry#
```

测试

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>

int main() {
    int fd;
    const char *path = "/proc/sys/vm/overcommit_memory";
    int flags = O_RDONLY;
    // Use openat to open the file
    fd = openat(AT_FDCWD, path, flags);
    if (fd == -1) {
        perror("openat failed");
        return 1;
    }
    // Check if openat opened the correct file
    if (fd > 0) {
        printf("openat returned file descriptor: %d\n", fd);

        // Check if the file descriptor value is '0'
        if (fd == '0') {
            printf("openat returned correct value: %d\n", fd);
        } else {
            printf("openat returned incorrect value: %d\n", fd);
        }
        // Close the file descriptor after use
        close(fd);
    } else {
        printf("openat returned incorrect value: %d\n", fd);
    }
```

```
    return 0;

}
```

```
        7:
        7:     initialize program: ./oom_openat
        7:
        7:
        7:     transferring control: ./oom_openat
        7:
[ 11.506047 0:8 linux_syscall_api::syscall_fs::imp::io:391] p: /proc/sys/vm/overcommit_memory
[ 11.507259 0:8 linux_syscall_api::syscall_fs::imp::io:393] Opening file: /proc/sys/vm/overcommit_memory
        7:
        7:     calling fini: ./oom_openat [0]
        7:
openat returned file descriptor: 48
openat returned correct value: 48
[ 11.512705 0:6 linux_syscall_api::syscall_fs::imp::io:391] p: /etc/passwd
[ 11.513865 0:6 linux_syscall_api::syscall_fs::imp::io:396] Unsupported path: /etc/passwd
```

经验证，识别到打开文件为/proc/sys/vm/overcommit_memory时，立即返回 '0' 的ASCII值