Đồ án 2 (Tiếp theo đồ án 1). Syscall về quản lý hệ thống tập tin

Chú ý: Các bạn phải code đồ án này trên code của đồ án 1.

1. Cài đặt system call **int CreateFile(char *name)**. CreateFile system call sẽ sử dụng Nachos FileSystem Object để tạo một file rỗng. Chú ý rằng filename đang ở trong user space, có nghĩa là buffer mà con trỏ trong user space trỏ tới phải được chuyển từ vùng nhớ user space tới vùng nhớ system space. System call CreateFile trả về 0 nếu thành công và -1 nếu có lỗi.

2. Cài đặt system call **OpenFileID Open(char *name, int type)** và **int Close(OpenFileID id).** User program có thể mở 2 loại file, file chỉ đọc và file đọc và ghi. Mỗi tiến trình sẽ được cấp một bảng mô tả file với kích thước cố định. Đồ án này, kích thước của bảng mô tả file là có thể lưu được đặc tả của 10 files. Trong đó, 2 phần tử đầu, ô 0 và ô 1 để dành cho console input và console output. System call mở file phải làm nhiệm vụ chuyển đổi địa chỉ buffer trong user space khi cần thiết và viết hàm xử lý phù hợp trong kernel. Bạn sẽ phải dùng đối tượng filesystem trong thư mục **filesys**. System call **Open** sẽ trả về id của file (OpenFileID = một số nguyên), hoặc là -1 nếu bị lỗi. Mở file có thể bị lỗi như trường hợp là không tồn tại tên file hay không đủ ô nhớ trong bảng mô tả file. Tham số *type* = 0 cho mở file đọc và ghi, = 1 cho file chỉ đọc. Nếu tham số truyền bị sai thì system call phải báo lỗi. System call sẽ trả về -1 nếu bị lỗi và 0 nếu thành công.

3. Cài đặt system call **int Read(char *buffer, int charcount, OpenFileID id)** và **int Write(char *buffer, int charcount, OpenFileID id)**. Các system call đọc và ghi vào file với id cho trước. Bạn cần phải chuyển vùng nhớ giữa user space và system space, và cần phải phân biệt giữa Console IO (OpenFileID 0, 1) và File. Lệnh Read và Write sẽ làm việc như sau: Phần console read và write, bạn sẽ sử dụng lớp SynchConsole. Được khởi tạo qua biến toàn cục gSynchConsole(bạn phải khai báo biến này). Bạn sẽ sử dụng các hàm mặc định của SynchConsole để đọc và ghi, tuy nhiên bạn phải chịu trách nhiệm trả về đúng giá trị cho user. Đọc và ghi với Console sẽ trả về số bytes đọc và ghi thật sự, chứ không phải số bytes được yêu cầu. Trong trường hợp đọc hay ghi vào console bị lỗi thì trả về -1. Nếu đang đọc từ console và chạm tới cuối file thì trả về -2. Đọc và ghi vào console sẽ sử dụng dữ liệu ASCII để cho input và output, (ASCII dùng kết thúc chuỗi là NULL (\0)). Phần đọc, ghi vào file, bạn sẽ sử dụng các lớp được cung cấp trong file system. Sử dụng các hàm mặc định có sẵn của filesystem và thông số trả về cũng phải giống như việc trả về trong synchconsole. Cả read và write trả số kí tự đọc, ghi thật sự. Cả Read và Write trả về -1 nếu bị lỗi và -2 nếu cuối file. Cả Read và Write sử dụng dữ liệu binary.

4. Cài đặt system call **int Seek(int pos, OpenFileID id).** Seek sẽ phải chuyển con trỏ tới vị trí thích hợp. *pos* lưu vị trí cần chuyển tới, nếu pos = -1 thì di chuyển đến cuối file. Trả về vị trí thực sự trong file nếu thành công và -1 nếu bị lỗi. Gọi Seek trên console phải báo lỗi.

5. Viết chương trình **createfile** để kiểm tra system call CreateFile. Bạn sẽ dùng tên file cố định, hoặc cho người dùng nhập vào từ console nếu như phần console IO của bạn là chạy được.

6. Viết chương trình **echo**, mỗi khi nhập một dòng từ console thì console xuất lại dòng đó

7. Viết chương trình **cat ,** yêu cầu nhập vô filename, rồi hiển thị nội dung của file đó

8. Viết chương trình **copy**, yêu cầu nhập tên file nguồn và file đích và thực hiện copy

9. Viết chương trình **reverse**, yêu cần nhập tên file nguồn và file đích, đọc nội dung file nguồn, đảo nghịch nội dung và ghi vào file đích.

10. Viết bất cứ chương trình nào khác mà bạn nghĩ là cần thiết để chứng minh giải pháp của bạn là đúng đắn. (mà chúng tôi chưa kiểm tra trong phần 5 tới 9)

<u>Yêu cầu nộp bài:</u> MSSV1_MSSV2_MSSV3.rar

- Mã nguồn
- Báo cáo:
  - Thiết kế hệ thống quản lý tập tin.
  - Viết tiếp với báo cáo đồ án 1.