KHOA CÔNG NGHỆ THÔNG TIN

ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

MÔN HỆ ĐIỀU HÀNH

Project 02: Các thao tác với files

ĐỒ ÁN/BÀI TẬP MÔN HỌC HỆ ĐIỀU HÀNH

HỌC KỲ I – NĂM HỌC 2022 - 2023

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện Thoại: (08) 38.354.266 - Fax:(08) 38.350.096

Danh sách thành viên nhóm:

STT	MSSV	Họ và tên	Emai
1	20120049	Nguyễn Hải Đăng	20120049@student.hcmus.edu.vn
2	20120077	Nguyễn Quang Hiển	20120077@student.hcmus.edu.vn
3	20120084	Nguyễn Văn Hiếu	20120084@student.hcmus.edu.vn

Phân công và đánh giá mức độ hoàn thành:

Phần	Câu	Ghi chú	Phân công	Điểm	Mức độ hoàn thành
1	1	Create	Đăng	0.75	100%
	2	Open	Đăng	0.75	100%
	3	Close	Đăng	0.75	100%
	4	Read	Đăng	0.75	100%
	5	Write	Hiếu	0.75	100%
	6	Seek	Hiếu	0.75	100%
	7	Remove	Hiếu	0.75	100%
	8	Sao chép vùng nhớ kernel-user	Hiếu	0.75	100%
2	1	createfile	Hiển	0.5	100%
	2	open	Hiển	0.5	100%
	3	copy	Hiển	0.5	100%
	4	delete	Hiển	0.5	100%
	5	concatenate	Hiển	0.5	100%
		Không để user làm sụp hệ điều hành	Hiển	0.5	100%
		Báo cáo	Đăng	1	100%

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện Thoại: (08) 38.354.266 - Fax:(08) 38.350.096

Mục lục

I. Cài đặt System call thao tác với file 3

1. System call int Create(char* name) 3

2. System call OpenFileId Open(char* name) 3

3. System call int Close(OpenFileId id) 4

4. System call int Read(char* buffer, int size, OpenFileId id) và 4

int Write(char* buffer, int size, OpenFileId id) 4

6. System Call int Seek(int position, OpenFileId id) 6

7. System Call int Remove(char* name) 6

II. Viết chương trình người dùng 6

1. Chương trình createfile 6

2. Chương trình cat 7

3. Chương trình copy 8

4. Chương trình delete 9

5. Chương trình concatenate 11

IV. Tài liệu tham khảo 12

I. Cài đặt System call thao tác với file

1. System call int Create(char* name)

- Mô tả cài đặt: handleSC_Create()
- Chức năng: sử dụng Nachos FileSystem Object để tạo một file rỗng
- Cách viết:
  - Đầu tiên ta lấy dữ liệu được lưu tại thanh ghi r4
  - Filename đang được lưu ở user space, sử dụng stringUser2System để chuyển tới vùng nhớ System space
  - Sử dụng hàm SysCreateFile để kiểm tra tạo file thành công hay không
  - Tạo file thành công: trả về giá trị 0 ở thanh ghi r2
  - Tạo file thất bại: trả về giá trị -1 ở thanh ghi r2
  - Tăng Program Counter
- Các hàm hỗ trợ:
  - bool SysCreateFile(char* filename): ở hàm này sẽ kiểm tra độ dài filename nhập vào và sử dụng hàm Create có sẵn ở fileSystem để kiểm tra tạo file thành công hay không. Trả về True nếu thành công, False nếu thất bại

2. System call OpenFileId Open(char* name)

User program có thể mở hai loại file, file chỉ đọc và file đọc ghi. Mỗi tiến trình sẽ được cấp một bảng mô tả file với kích thước cố định. Hai phần tử đầu, ô 0 và ô 1 để dành cho console input và console output.

- Mô tả cài đặt: handleSC_Open()
- Chức năng: sử dụng Nachos FileSystem Object mở file
- Cách viết:
  - Đầu tiên ta lấy dữ liệu được lưu tại thanh ghi r4
  - Filename đang được lưu ở user space, sử dụng stringUser2System để chuyển tới vùng nhớ System space
  - Lấy giá trị kiểu đọc file lưu tại thanh ghi r5
  - Sử dụng hàm SysOpen để kiểm tra tạo file thành công hay không
  - Tăng Program Counter
- Các hàm hỗ trợ:

- int SysCreate(char* filename, int type): ở hàm này sẽ kiểm tra giá trị kiểu đọc file, nếu type khác 0 và 1 thì trả về -1 cho thanh ghi r2. Biến id là id của file, nếu id = -1 tức là có lỗi trong việc mở file, trả về -1 cho thanh ghi r2. Trả về id nếu mở file thành công.

3. System call int Close(OpenFileId id)

- Mô tả cài đặt: handleSC_Close()
- Chức năng: sử dụng Nachos FileSystem Object đóng một file với tham số truyền vào là ID của file
- Cách viết:
  - Đầu tiên ta lấy ID của file được lưu tại thanh ghi r4
  - Sử dụng hàm SysClose để kiểm tra tạo file thành công hay không
  - Đóng file thành công: trả về giá trị 0 ở thanh ghi r2
  - Đóng file thất bại: trả về giá trị -1 ở thanh ghi r2
  - Tăng Program Counter
- Các hàm hỗ trợ:
  - int SysClose(int id): sử dụng phương thức Open được xây dựng sẵn trong fileSystem.

4. System call int Read(char* buffer, int size, OpenFileId id) và int Write(char* buffer, int size, OpenFileId id)

Các System call đọc và ghi vào file với id cho trước. Phải chuyển vùng nhớ giữa user space và system space, cần phân biệt giữa Console IO (OpenFileID 0, 1) và File.

❖ **Read(char* buffer, int size, OpenFileId id)**

- Mô tả cài đặt: handleSC_Read()
- Chức năng: sử dụng Nachos FileSystem Object đọc một file với tham số truyền vào là ID của file, buffer là chuỗi chứa nội dung file, len chứa độ dài file, kết quả trả về là -1 nếu lỗi, -2 nếu thành công.
- Cách viết:
  - Đầu tiên ta lấy địa chỉ của tham số buffer được lưu tại thanh ghi r4, tham số countChar được lưu ở thanh ghi r5, và fileID ở thanh ghi r6.
  - Gọi hàm stringUser2System(virtAddress, countChar) để sao chép dữ liệu từ vùng nhớ User sang vùng nhớ của System.

- Tiến hành đọc nội dung của file vào buff bằng hàm sysRead, lưu ở thanh ghi r2.
- Sau đó, gọi hàm stringSys2User(buff, virtAddress, countChar) để sao chép buff chuỗi từ vùng nhớ system sang vùng nhớ user để người dùng sử dụng.
- Delete buffer, tăng program counter.

- Các hàm hỗ trợ:
  - int SysRead(char* buff, int countChar, int fId)
    - Nếu fID = 0: thì chưa có kí tự '\0' ở cuối chuỗi để kết thúc việc nhập chuỗi => cứ nhập chuỗi mà không thể dừng.
    - Nếu không thì: chuỗi đang có kí tự '\0' ở cuối và tiến hành đọc file.

❖ **int Write(char* buffer, int size, OpenFileId id)**

- Mô tả cài đặt: handleSC_Write()
- Chức năng: sử dụng Nachos FileSystem Object đọc một file với tham số truyền vào là ID của file, buffer là chuỗi chứa nội dung file, len chứa độ dài file, kết quả trả về là -1 nếu lỗi, -2 nếu thành công.
- Cách viết:
  - Đầu tiên ta lấy địa chỉ của tham số buffer được lưu tại thanh ghi r4, tham số countChar được lưu ở thanh ghi r5, và fileID ở thanh ghi r6.
  - Gọi hàm stringUser2System(virtAddress, countChar) để sao chép dữ liệu từ vùng nhớ User sang vùng nhớ của System.
  - Tiến hành ghi nội dung của buff vào file bằng hàm sysWrite, lưu ở thanh ghi r2.
  - Sau đó, gọi hàm stringSys2User(buff, virtAddress, countChar) để sao chép chuỗi từ vùng nhớ system sang vùng nhớ user để người dùng sử dụng.
  - Delete buffer, tăng program counter.
- Các hàm hỗ trợ:
  - int SysWrite(char* buff, int countChar, int fId): hàm kiểm tra file có hợp lệ không
    - Nếu fID = 1: thì chưa có kí tự '\0' ở cuối chuỗi để kết thúc việc nhập chuỗi => cứ nhập chuỗi mà không thể dừng.

- Nếu không thì: chuỗi đang có kí tự '\0' ở cuối và tiến hành đọc file.

6. System Call int Seek(int position, OpenFileId id)

Seek sẽ phải chuyển con trỏ tới vị trí thích hợp. Pos lưu vị trí cần chuyển tới, nếu position = -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.

7. System Call int Remove(char* name)

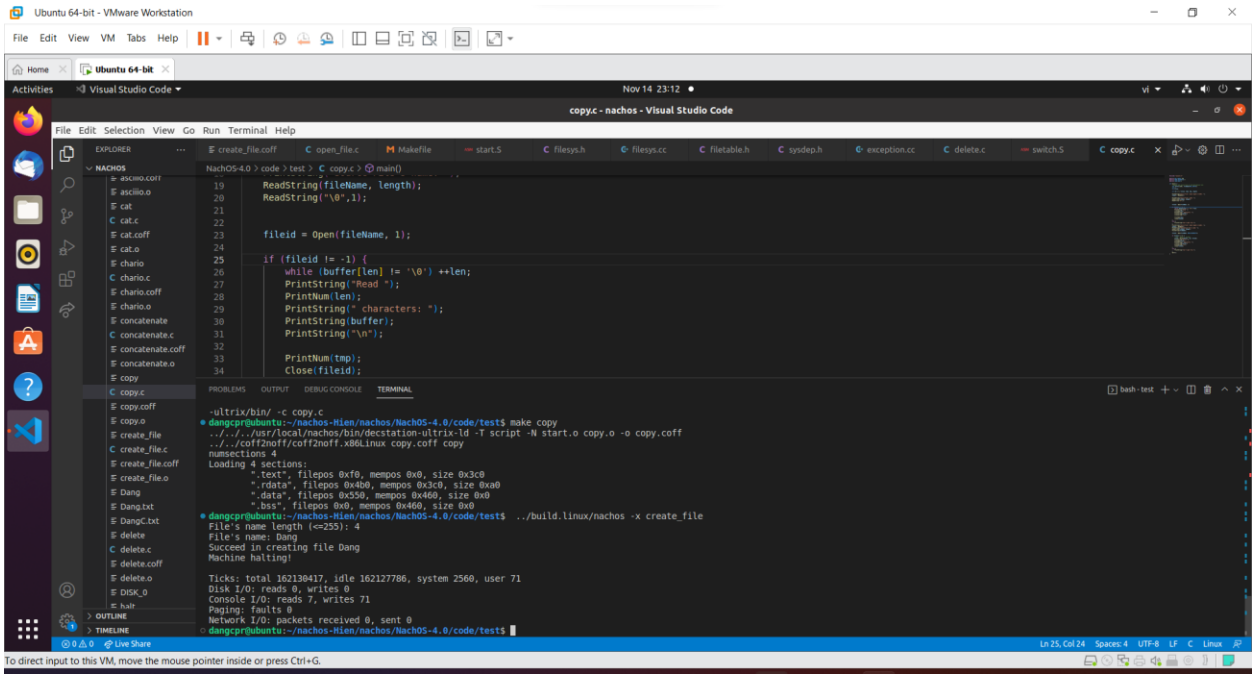
Remove system call sẽ sử dụng Nachos FileSystem Object để xóa file.

- Mô tả cài đặt: handleSC_Remove()
- Cách viết:
  - Đầu tiên ta lấy dữ liệu được lưu tại thanh ghi r4
  - Filename đang được lưu ở user space, sử dụng stringUser2System để chuyển tới vùng nhớ System space
  - Sử dụng hàm SysRemoveFile để kiểm tra xóa file thành công hay không
  - Xóa file thành công: trả về giá trị 1 ở thanh ghi r2
  - Xóa file thất bại: trả về giá trị -1 ở thanh ghi r2
  - Tăng Program Counter
- Các hàm hỗ trợ:
  - int SysRemoveFile(char* filename): đầu tiên tiến hành mở file để kiểm tra file có tồn tại hay không. File không tồn tại thì trả về -1. File có tồn tại, tiến hành xóa file, sử dụng hàm Remove có sẵn ở fileSystem.

II. Viết chương trình người dùng

1. Chương trình createfile

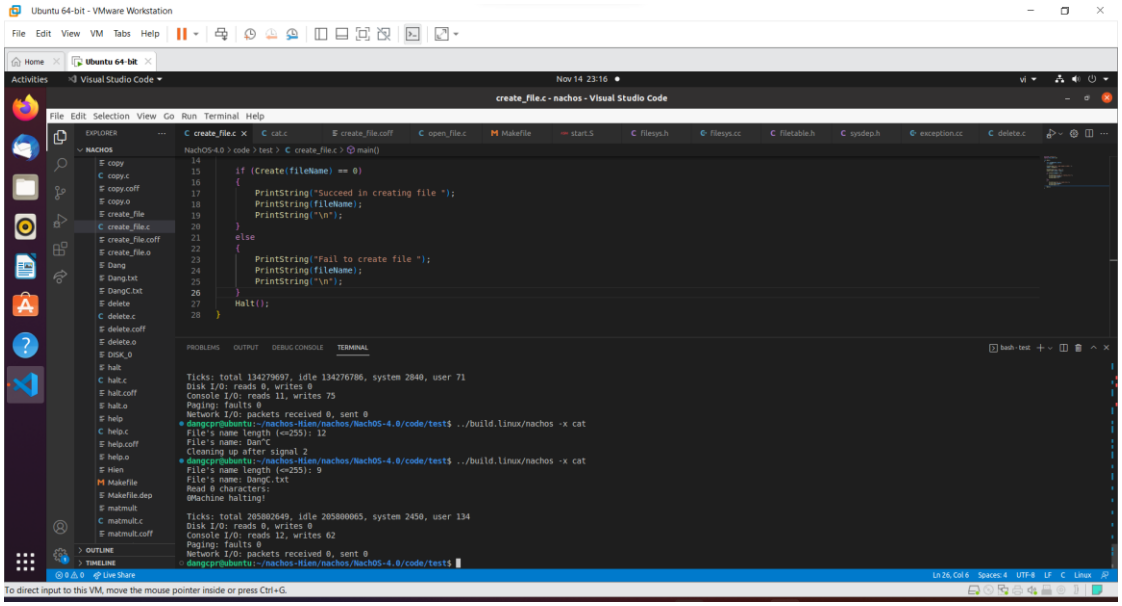
- ❖ Người dùng nhập độ dài tên file(syscall Readnum) cần tạo rồi sau đó nhập tên file(syscall ReadString).
- ❖ Gọi system call Create(filename), kết quả trả về bằng 0 thì file đã được tạo thành công. Ngược lại tạo file thất bại.
- ❖ Thông báo cho người dùng và gọi hàm Halt() để kết thúc.



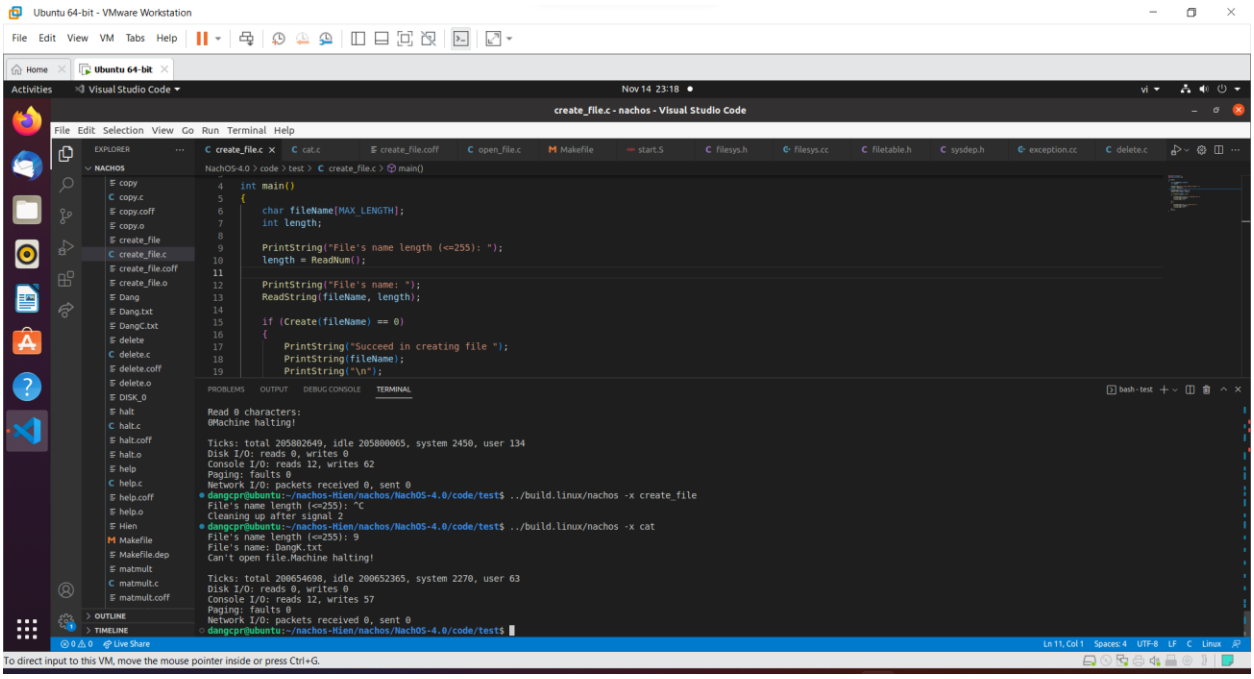
Hình 1. Tạo file thành công.

2. Chương trình cat

- Người dùng nhập độ dài tên file(syscall Readnum) cần tạo rồi sau đó nhập tên file(syscall ReadString).
- Sau đó gọi system call Open để mở file đó với type bằng 1(chỉ đọc)
- Lấy kích thước thực sự của file lưu vào biến len. Sử dụng system call PrinString để in dữ liệu.



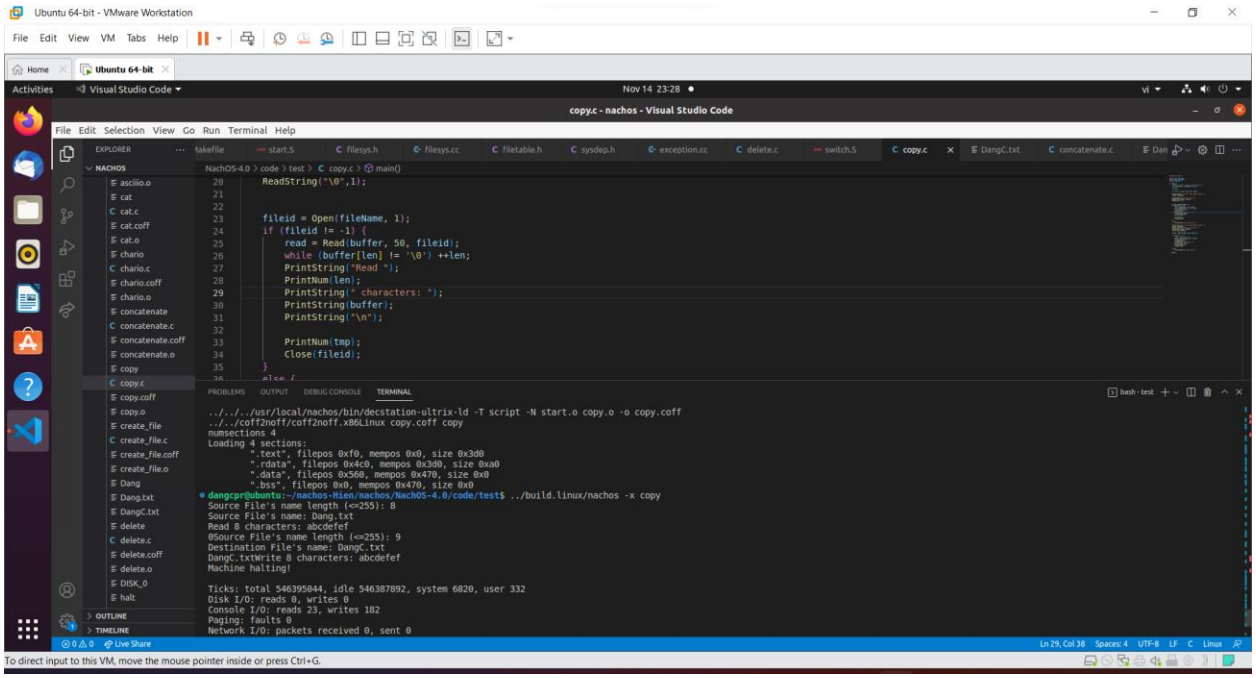
Hình 2. Đọc file thành công.



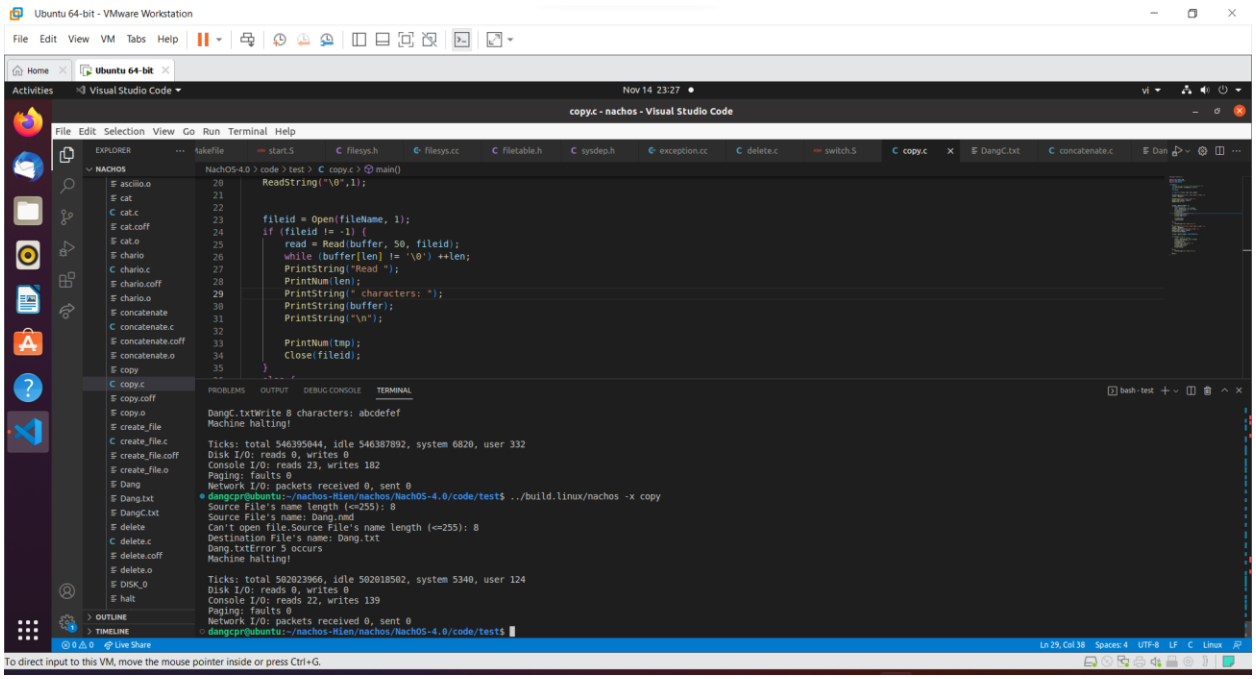
Hình 3. Đọc file thất bại.

3. Chương trình copy

- Người dùng nhập độ dài tên file nguồn (syscall Readnum) rồi sau đó nhập tên file (syscall ReadString).
- Sau đó gọi system call Open để mở file đó với type bằng 1(chỉ đọc).
- Nội dung file và độ dài file sẽ được lưu vô buffer và len.
- Tiếp theo, người dùng nhập độ dài tên file đích (syscall Readnum) cần copy nội dung rồi sau đó nhập tên file (syscall ReadString).
- Sau đó gọi system call Open để mở file đó với type bằng 0 (đọc và ghi).
- Nếu mở file thành công, system call write được gọi với nội dung write đã được lưu trong buffer và độ dài len.
- Sau đó, file đích đã được ghi nội dung từ file nguồn.



Hình 4. Copy thành công.

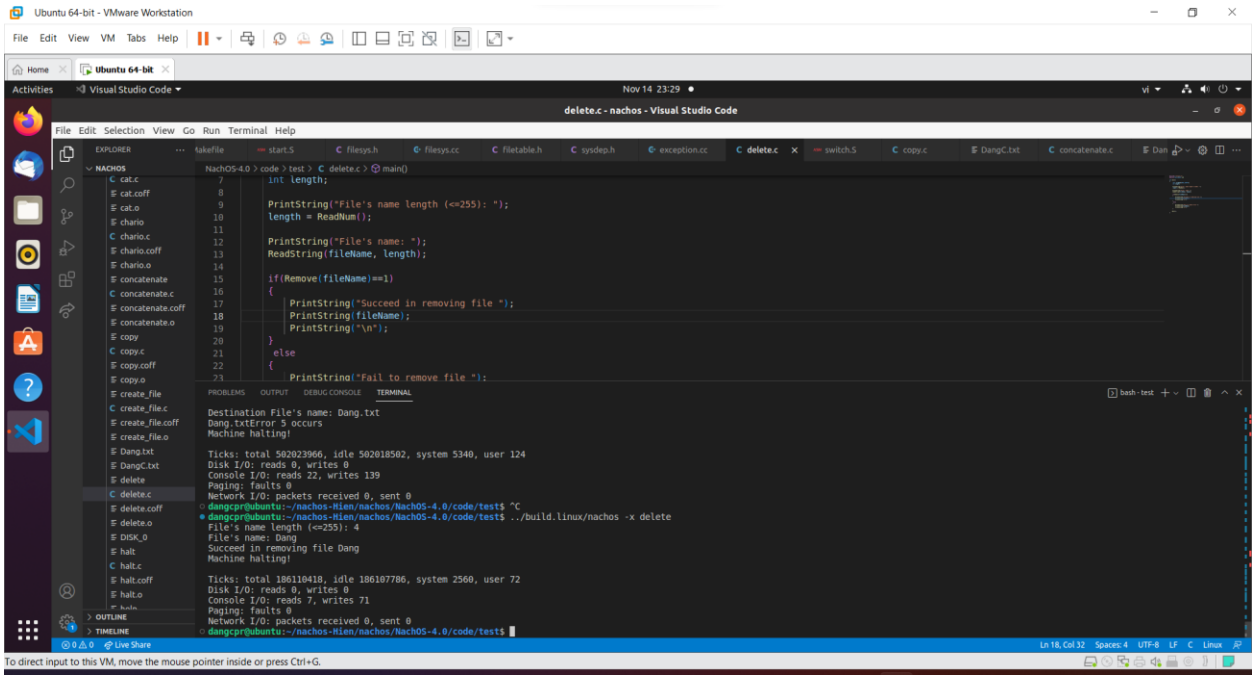


Hình 5. Đọc file thất bại

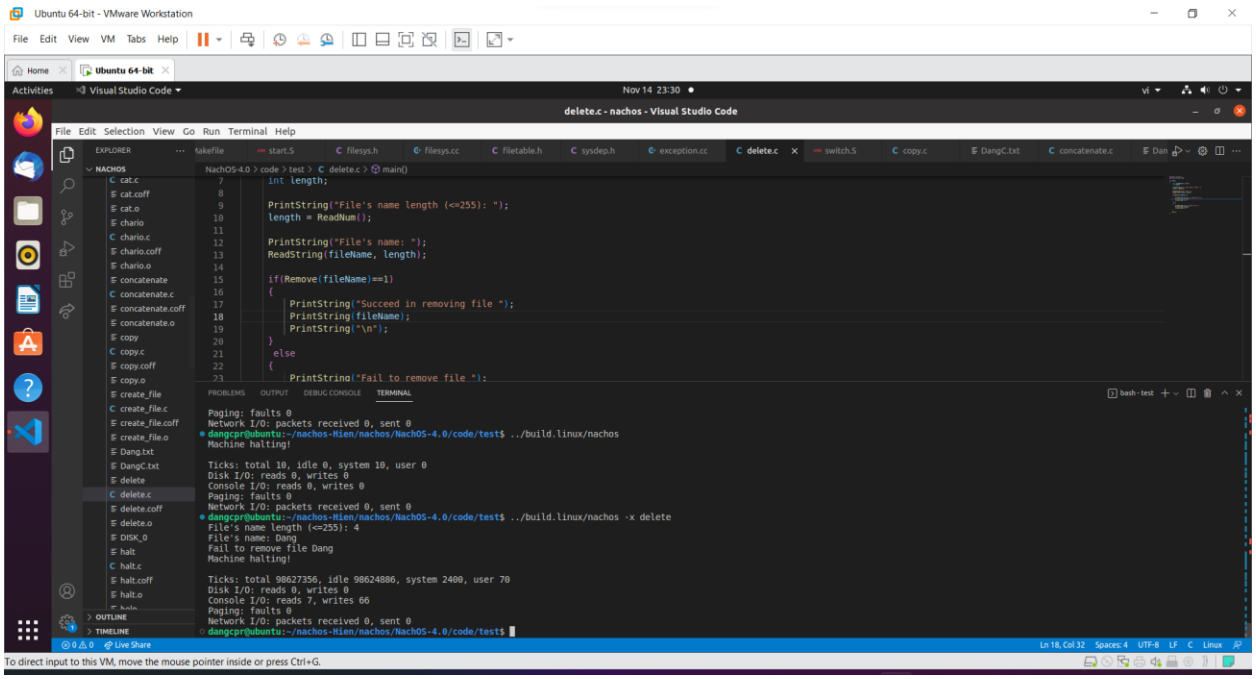
4. Chương trình delete

- Người dùng nhập độ dài tên file(syscall Readnum) cần tạo rồi sau đó nhập tên file(syscall ReadString).

- Gọi system call Remove(filename), kết quả trả về bằng 1 thì file đã được xóa thành công. Ngược lại xóa file thất bại.
- Thông báo cho người dùng và gọi hàm Halt() để kết thúc.



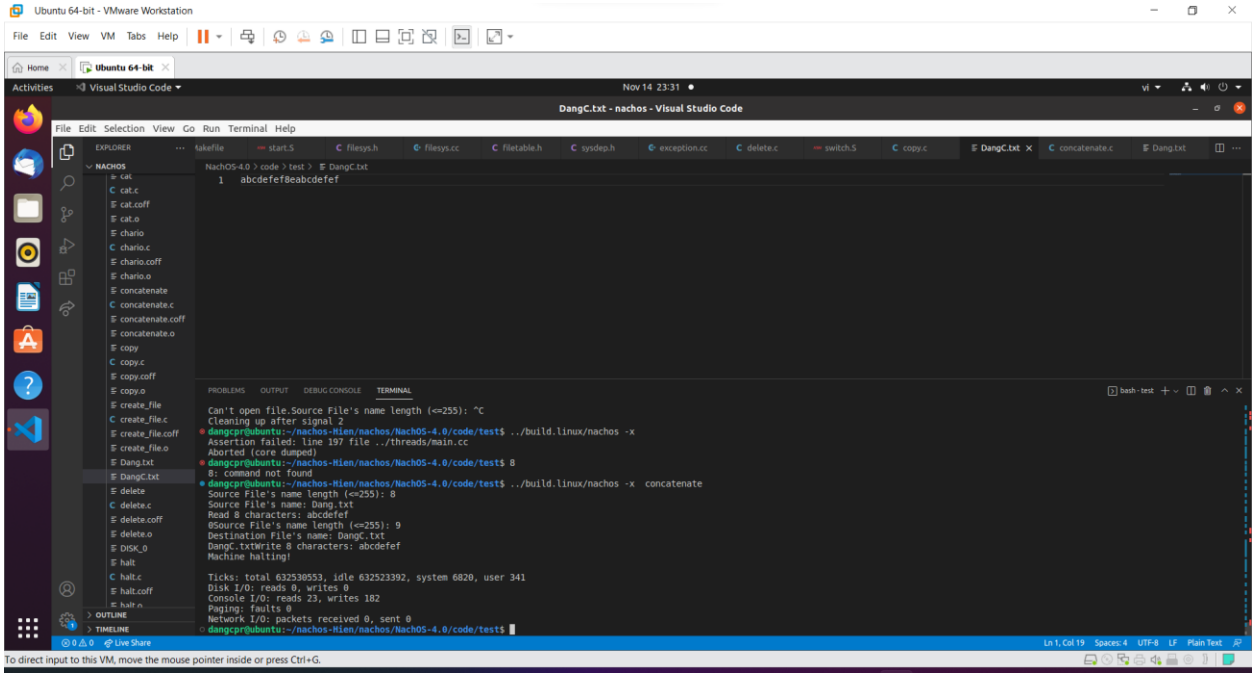
Hình 6. Xóa file thành công.



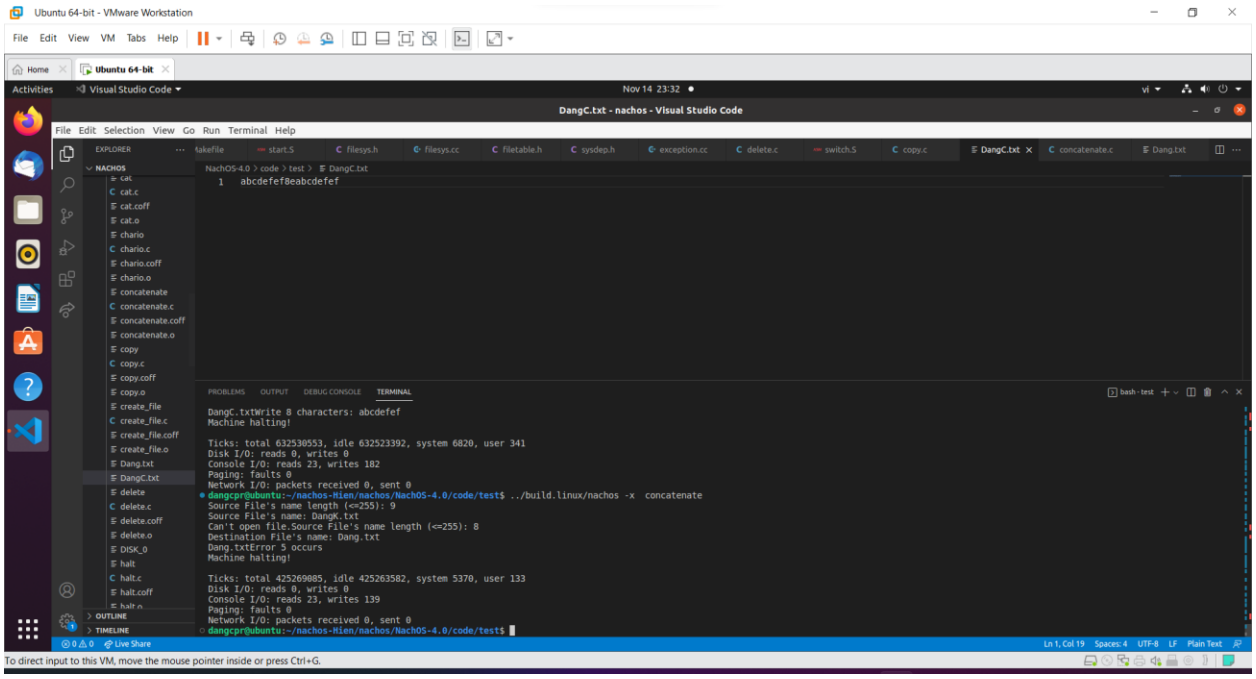
Hình 7. Xóa file thất bại.

5. Chương trình concatenate

- Người dùng nhập độ dài tên file nguồn (syscall Readnum) rồi sau đó nhập tên file (syscall ReadString).
- Sau đó gọi system call Open để mở file đó với type bằng 1(chỉ đọc).
- Nội dung file và độ dài file sẽ được lưu vô buffer và len.
- Tiếp theo, người dùng nhập độ dài tên file đích (syscall Readnum) cần copy nội dung rồi sau đó nhập tên file (syscall ReadString).
- Sau đó gọi system call Open để mở file đó với type bằng 0 (đọc và ghi).
- Nếu mở file thành công, thì system call seek sẽ được gọi với pos = -1 (trỏ về cuối file)
- Sau đó, system call write được gọi với nội dung write đã được lưu trong buffer và độ dài len.
- Sau đó, file đích đã được nối nội dung từ file nguồn.



Hình 8. Nối file thành công.



Hình 9. Nối file thất bại.

IV. Tài liệu tham khảo

[1] leduythuccs/nachos-project: NACHOS - Not Another Completely Heuristic Operating System (github.com)

[2] nguyenthanhchungfit/Nachos-Programing-HCMUS: This project includes: (github.com)