



**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN - ĐHQG  
TP.HCM  
VNUHCM - UNIVERSITY OF SCIENCE**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN THỰC HÀNH HỆ ĐIỀU HÀNH:**

**HỆ ĐIỀU HÀNH NACHOS**

**MÃ MÔN HỌC: CSC-10007\_21CLC04**

**THỰC HIỆN: Nhóm 18, Lớp 21CLC04**

**GVHD:**

**LÊ GIANG THANH**

**NGUYỄN THANH QUÂN**

**LÊ HÀ MINH**

**Tp. Hồ Chí Minh, tháng 4 năm 2023**

# BIÊN BẢN PHÂN CÔNG CÔNG VIỆC NHÓM

## I. Danh sách thông tin nhóm

**Tên nhóm: nhóm 18**

ST T	MSSV	Họ và tên	Ghi chú
1	21127627	Cao Nguyễn Khánh	
2	21127711	Trịnh Minh Trung	Nhóm trưởng
3	21127535	Thành Thiện Nhân	

## Phân công nhiệm vụ

STT	Mô tả	Thực hiện	Mức độ hoàn thành
1.1	Syscall Create	Trịnh Minh Trung	100%
1.2	Syscall OpenFileID, Close	Trịnh Minh Trung	100%
1.3	Syscall Read, Write	Cao Nguyễn Khánh	100%
1.4	Syscall Seek	Thành Thiện Nhân	100%
1.5	Syscall Remove	Trịnh Minh Trung	100%
2.1	Syscall SocketTCP	Cao Nguyễn Khánh	100%
2.2	Syscall Connect	Cao Nguyễn Khánh	100%
2.3	Syscall Send	Cao Nguyễn Khánh	100%

2.4	Syscall Receive	Cao Nguyễn Khánh	100%
3	Advanced	Cao Nguyễn Khánh	50%
4.1	Test program Create	Trịnh Minh Trung	100%
4.2	Test program Copy	Thành Thiện Nhân	100%
4.3	Test program Cat	Thành Thiện Nhân	100%
4.4	Test program Delete	Trịnh Minh Trung	100%
4.5	Test program Concatenate	Thành Thiện Nhân	100%
4.6	Test program Echo	Cao Nguyễn Khánh	100%
4.7	Test program File Transfer	Thành Thiện Nhân	100%
5	Report	Nhóm 18	100%

**Mức độ hoàn thành đồ án = 95%**

# MỤC LỤC

## Mục lục

<b>IMPLEMENT SYSTEMCALL FOR FILE OPERATION</b>	<b>4</b>
1. SystemCall Create	4
2. SystemCall Open	4
3. SystemCall Close	5
4. SystemCall Read	5
5. SystemCall Write	6
6. SystemCall Seek	6
7. SystemCall Remove	7
8. SystemCall SocketTCP	7
9. SystemCall Connect	7
10. SystemCall Send	8
11. SystemCall Receive	9
12. SystemCall CloseSocket (giống như SC_Close )	9
<b>TEST PROGRAM</b>	<b>10</b>
1. Chương trình createfile	10
2. Chương trình cat	10
3. Chương trình copy	11
4. Chương trình delete	12
5. Chương trình concatenate	13
6. Chương trình echo	14
7. Chương trình filetransfer	16

# NỘI DUNG BÁO CÁO

## IMPLEMENT SYSTEMCALL FOR FILE OPERATION

### 1. SystemCall Create

- Mô tả cài đặt **SC\_Create**:
  - Input: char\* filename.
  - Output: Thành công: 0; Lỗi: -1.
  - Used: Tạo 1 file rỗng.
- Ý tưởng:
  - Chuyển dữ liệu filename từ user to system bằng hàm User2System()
  - kiểm tra xem filename hợp lệ không, nếu không → báo lỗi, nếu hợp lệ tạo file mới nếu thành công return 0, nhưng nếu không đủ vùng nhớ hoặc tạo gặp lỗi thì báo lỗi return -1.

### 2. SystemCall Open

- Mô tả cài đặt **SC\_Open**:
  - Input: char\* filename, int type.
  - Output: Thành công: 0, Không thành công: -1.
  - Used: Mở file.
- Ý tưởng:
  - Đọc các tham số từ các thanh ghi 4, 5 tương ứng với filenameAddr, fileType.
  - Chuyển dữ liệu filename từ user to system bằng hàm User2System()
  - Với type là cách mở (type = 0: Mở để đọc và ghi; type = 1: mở để đọc).
  - Kiểm tra filename hợp lệ hay không.
  - Sau đó tiến hành mở file, nếu mở được thì tiếp tục tùy chọn cách mở file dựa trên type của đầu vào.

### 3. SystemCall Close

- Mô tả cài đặt **SC\_Close**:
  - Input: fileID.
  - Output: Thành công: 0, Không thành công: -1.
  - Used: Đóng file.
- Ý tưởng:
  - Đọc các tham số từ các thanh ghi 4 tương ứng với OpenFileId fileId. (typedef int OpenFileId;)
  - Nếu fileId < 0 hoặc fileId > số lượng file đã được định nghĩa ban đầu (N = 20) thì đóng không thành công.
  - Trong trường hợp fileID thỏa mãn thì cho mảng file descriptor tại vị trí fileId là NULL, nếu không xóa fileID ra khỏi bảng des được thì đóng file không thành công.

### 4. SystemCall Read

- Mô tả cài đặt **SC\_Read**:
  - Input: char\* buffer, int size, OpenFileID id.
  - Output: Số lượng kí tự đọc thành công.
  - Used: Đọc các kí tự trong file vào buffer với kích thước size.
- Ý tưởng:
  - Đọc các tham số từ các thanh ghi 4, 5 và 6 tương ứng với bufferAddress, size, và fileID.
  - Xét fileID, nếu fileID = 0 thì chuyển sang đọc input từ console; Nếu fileID = 1, đây là consoleOUTPUT nên ta không thể đọc → báo lỗi
  - Đọc nội dung file vào buffer, chuyển dữ liệu về lại bufferAddress qua hàm System2User ().
  - Trả về số byte đọc được nếu FileID hợp lệ, ngược lại báo lỗi.

### 5. SystemCall Write

- Mô tả cài đặt **SC\_Write**:
  - Input: char\* buffer, int size, OpenFileID id.

- Output: Số lượng kí tự ghi thành công.
- Used: Ghi các kí tự trong buffer vào file.
- **Ý tưởng:**
  - Đọc các tham gia số từ các thanh ghi 4, 5 và 6 tương ứng với bufferAddress, size, và fileID.
  - Chuyển dữ liệu bufferAddress từ user to system bằng hàm User2System().
  - Xét fileID, nếu fileID = 1 thì chuyển sang ghi input, Nếu fileID = 0, đây là consoleINPUT nên ta không thể ghi → báo lỗi.
  - Ghi nội dung buffer vào file, và trả về số byte ghi được nếu FileID hợp lệ, ngược lại báo lỗi.

## 6. SystemCall Seek

- **Mô tả cài đặt SC\_Seek:**
  - Input: int position, OpenFileID id.
  - Output: Vị trí trong file.
  - Used: Đặt con trỏ file tại vị trí position ở fileID = id.
- **Ý tưởng:**
  - Đọc từ register 4 và register 5 lần lượt là pos và fileID. Kiểm tra tính hợp lệ của fileID. Nếu không hợp lệ thì lỗi.
  - Kiểm tra nếu position = -1 thì trả về cuối file. Nếu position không nằm trong khoảng [0, lenFileSpace] thì lỗi và position = -1 (Trả về cuối file).
  - Nếu không thì đặt con trỏ file ở vị trí pos, trả ra pos

## 7. SystemCall Remove

- **Mô tả cài đặt SC\_Remove:**
  - Input: char\* filename.
  - Output: Thành công: 0, Lỗi: -1.
  - Used: Xóa 1 file.
- **Ý tưởng:**

- Chuyển dữ liệu filename từ system to user bằng hàm System2User().
- Kiểm tra xem filename có đang mở hay không, nếu đang mở thì lỗi, nếu không thì xóa file.

## 8. SystemCall SocketTCP

### ● Mô tả cài đặt SC\_SocketTCP:

- Input:
- Output: Thành công: vị trí của fileID trong mảng, Lỗi: -1.
- Used: tạo ra 1 socketID, và gán ID vào fileID trong mảng

### ● Ý tưởng:

- Sử dụng hàm socket(AF\_INET, SOCK\_STREAM, 0) để tạo một socket TCP mới.
- Lưu trữ thông tin socket mảng file descriptor để quản lý các socket. Lưu trữ vào biến “int file ” trong lớp OpenFile.
- Trả về file descriptor (index ) của socket nếu tạo socket thành công, hoặc -1 nếu gặp lỗi.

## 9. SystemCall Connect

### ● Mô tả cài đặt SC\_Connect:

- Input: int socketid (index của ID), char \*ip, int port
- Output: Thành công: 0, Lỗi: -1.
- Used: kết nối đến server theo địa chỉ IP và cổng port được chỉ định.

### ● Ý tưởng:

- Đọc các tham gia số từ các thanh ghi 4, 5 và 6 tương ứng với indexsocketID, virtAdr, port.
- Chuyển dữ liệu virtAdr từ user to system bằng hàm User2System()
- Lấy thông tin socket (ID) từ mảng file descriptor, thông qua indexsocketID.
- Sử dụng hàm connect(ID, ip, port) để kết nối đến server theo địa chỉ IP và cổng port được chỉ định.
- Nếu kết nối thành công, trả về 0, nếu kết nối thất bại, trả về -1.



## 10. SystemCall Send

- **Mô tả cài đặt SC\_Send:**

- Input: int socketid (index của ID), char \*buffer, int len
- Output: Thành công: Trả về số lượng byte đã gửi, Lỗi: -1.
- Used: gửi 1 mảng ký tự đến server.

- **Ý tưởng:**

- Đọc các tham số từ các thanh ghi 2, 4, 5 và 6 tương ứng với int connection, indexsocketID, virtAdr, len.
- Lấy thông tin socket (ID) từ mảng file descriptor thông qua indexsocketID.
- Chuyển dữ liệu virtAdr từ user to system bằng hàm User2System()
- Sử dụng hàm send(socketid, buffer, strlen(buffer), 0) trong thư viện socket.h để gửi dữ liệu từ buffer đến socket.
- Trả về số lượng byte đã gửi nếu thành công, hoặc trả về 0 nếu kết nối bị đóng hoặc -1 nếu gặp lỗi.

## 11. SystemCall Receive

- **Mô tả cài đặt SC\_Receive:**

- Input: int socketid (index của ID), char \*buffer, int len
- Output: Thành công: Trả về số lượng byte đã nhận, Lỗi: -1.
- Used: nhận 1 mảng ký tự đến từ server.

- **Ý tưởng:**

- Đọc các tham số từ các thanh ghi 2, 4, 5 và 6 tương ứng với int connection, indexsocketID, virtAdr, len.
- Lấy thông tin socket từ mảng file descriptor.
- Sử dụng hàm read(socketid, buffer, len) trong thư việnunistd.h để nhận dữ liệu từ server.
- Chuyển dữ liệu về lại virtAdr qua hàm System2User().
- Trả về số lượng byte đã gửi hoặc nhận nếu thành công, hoặc trả về 0 nếu kết nối bị đóng hoặc -1 nếu gặp lỗi.

## 12. SystemCall CloseSocket (giống như SC\_Close )

- Mô tả cài đặt **SC\_CloseSocket**:
  - Input: fileID.
  - Output: Thành công: 0, Không thành công: -1.
  - Used: Đóng file.
- Ý tưởng:
  - Đọc các tham số từ các thanh ghi 4 tương ứng với OpenFileId fileId. (typedef int OpenFileId;)
  - Nếu fileId < 0 hoặc fileId > số lượng file đã được định nghĩa ban đầu (N = 20) thì đóng không thành công.
  - Trong trường hợp fileID thỏa mãn thì cho mảng file descriptor tại vị trí fileId là NULL, nếu không xóa fileID ra khỏi bảng des được thì đóng file không thành công.

## TEST PROGRAM

### 1. Chương trình createfile

- Yêu cầu chương trình: Viết chương trình createfile để kiểm tra SystemCall Create. Nhập tên file từ console.
- Cách thức hoạt động:
  - Bước 1: Cho người dùng nhập vào tên file.
  - Bước 2: Tạo file với tên mới nhập
- Hình ảnh demo chương trình:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4...
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x createfile
Nhap ten file ban muon tao: demoHDH.txt

Shutdown, initiated by user program.Machine halting!

Ticks: total 314649055, idle 314647683, system 1340, user 32
Disk I/O: reads 0, writes 0
Console I/O: reads 12, writes 28
Paging: faults 0
Network I/O: packets received 0, sent 0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

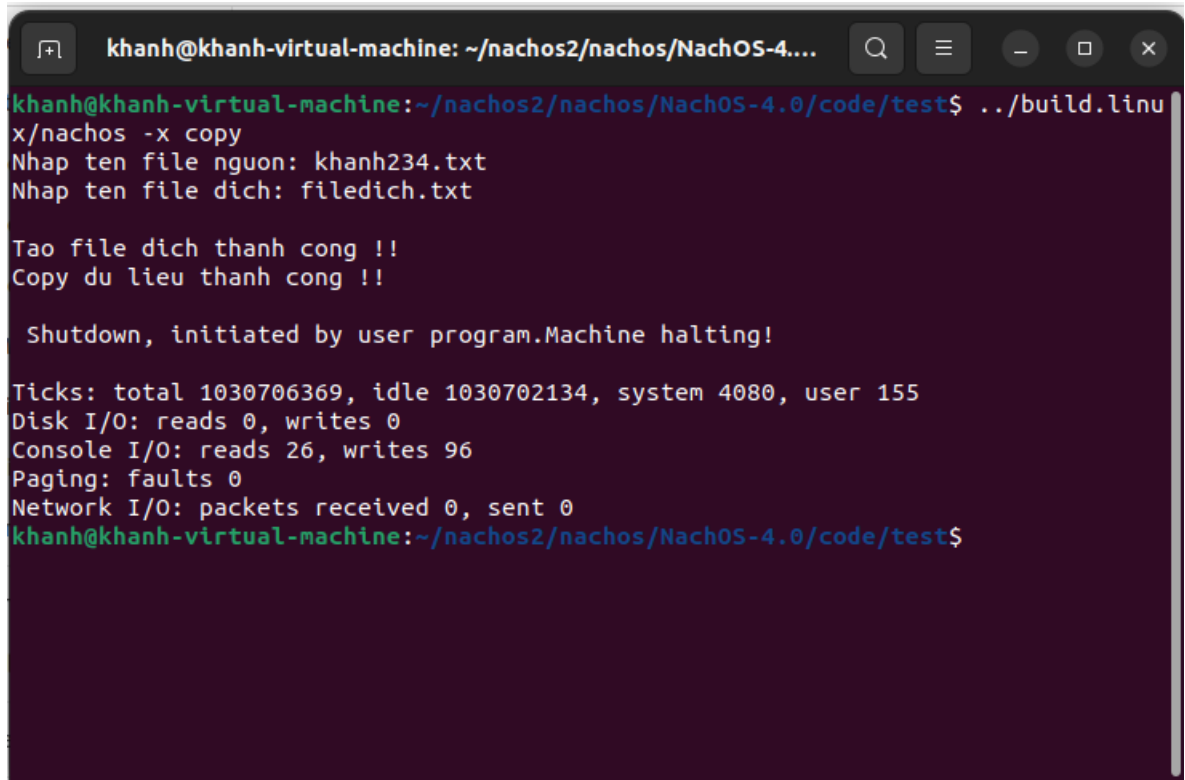
## 2. Chương trình cat

- Yêu cầu chương trình: Viết chương trình cat, yêu cầu nhập filename, rồi hiển thị nội dung của file đó.
- Cách thức hoạt động:
  - Bước 1: Nhập vào tên file.
  - Bước 2: Mở file và lấy độ dài của file bằng cách seek đến vị trí cuối file. Nếu mở file không thành công thì kết thúc.
  - Bước 3: Sau khi lấy được len thì ta đọc file với độ dài là của chuỗi đọc được là len và in ra màn hình.
- Hình ảnh demo chương trình:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....  
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x cat  
Nhap ten file ban muon doc: khanh234.txt  
  
Noi dung file la :  
mot hai ba bon  
nam sau bay tam  
chin muoi  
  
Shutdown, initiated by user program.Machine halting!  
  
Ticks: total 310136792, idle 310133262, system 3440, user 90  
Disk I/O: reads 0, writes 0  
Console I/O: reads 13, writes 90  
Paging: faults 0  
Network I/O: packets received 0, sent 0  
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

### 3. Chương trình copy

- Yêu cầu chương trình: Viết chương trình copy, yêu cầu nhập tên của file nguồn và file đích, sau đó thực hiện sao chép nội dung từ file nguồn và dán vào file đích.
- Cách thức hoạt động:
  - Bước 1: Cho người dùng nhập vào tên file nguồn và file đích.
  - Bước 2: Mở file nguồn và lấy độ dài của file bằng cách seek đến vị trí cuối file. Nếu mở file không thành công thì kết thúc.
  - Bước 3: Đọc file nguồn và lưu vào mảng char.
  - Bước 4: Sau đó ta sẽ mở file đích. Nếu không mở được file đích thì tạo một file đích mới với tên vừa nhập, nếu tạo không được thì dừng chương trình.
  - Bước 5: Ghi dữ liệu đã đọc từ file nguồn sang file đích rồi kết thúc chương trình.
- Hình ảnh demo chương trình:



```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4.0...
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x copy
Nhap ten file nguon: khanh234.txt
Nhap ten file dich: filedich.txt

Tao file dich thanh cong !!
Copy du lieu thanh cong !!

Shutdown, initiated by user program.Machine halting!

Ticks: total 1030706369, idle 1030702134, system 4080, user 155
Disk I/O: reads 0, writes 0
Console I/O: reads 26, writes 96
Paging: faults 0
Network I/O: packets received 0, sent 0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

#### 4. Chương trình delete

- Yêu cầu chương trình: Viết chương trình delete để kiểm tra SystemCall Remove.
- Cách thức hoạt động:
  - Bước 1: Cho người dùng nhập tên file muốn xóa
  - Bước 2: Xóa file và kết thúc chương trình.
- Hình ảnh demo chương trình:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4...
usr/local/nachos/decstation-ultrix/bin/ -I../userprog -I../lib -c -o delete.o d
elete.c
../../../../usr/local/nachos/bin/decstation-ultrix-ld -T script -N start.o delete.o
-o delete.coff
../../../../coff2noff/coff2noff.x86Linux delete.coff delete
numsections 4
Loading 4 sections:
    ".text", filepos 0xf0, mempos 0x0, size 0x220
    ".rdata", filepos 0x310, mempos 0x220, size 0x20
    ".data", filepos 0x330, mempos 0x240, size 0x0
    ".bss", filepos 0x0, mempos 0x240, size 0x0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linu
x/nachos -x delete
Nhập tên file bạn muốn xóa: demoHDH.txt

Shutdown, initiated by user program.Machine halting!

Ticks: total 1686404255, idle 1686402883, system 1340, user 32
Disk I/O: reads 0, writes 0
Console I/O: reads 12, writes 28
Paging: faults 0
Network I/O: packets received 0, sent 0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

## 5. Chương trình concatenate

- Yêu cầu chương trình: Viết chương trình concatenate để nối nội dung của 2 file, yêu cầu nhập tên file nguồn 1 và file nguồn 2.
- Cách thức hoạt động:
  - Bước 1: Cho người dùng nhập vào tên file nguồn 1 và file nguồn 2.
  - Bước 2: Mở lần lượt file nguồn 1 và file nguồn 2, nếu mở không được 1 trong 2 file thì báo lỗi và kết thúc chương trình. Nếu mở được cả 2 file thì đọc dữ liệu trong từng file.
  - Bước 3: Tạo file đích, nếu tạo thất bại thì báo lỗi và kết thúc chương trình.
  - Bước 4: Nếu tạo thành công, dán lần lượt từng nội dung của file nguồn 1 và file nguồn 2 vào file đích. Sau đó kết thúc chương trình.
- Hình ảnh demo chương trình:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x concatenate
Nhap ten file nguon 1: khanh24.txt
Nhap ten file nguon 2: filedich.txt

Noi file thanh cong

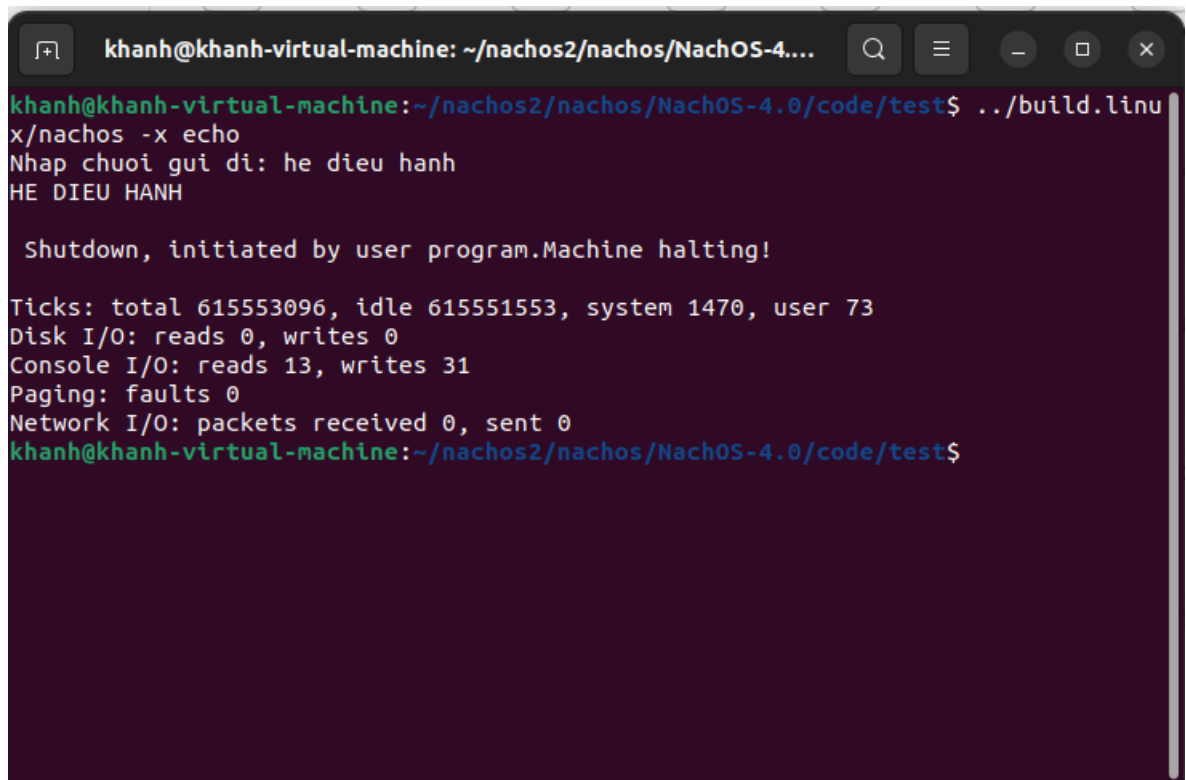
Shutdown, initiated by user program.Machine halting!

Ticks: total 1401352681, idle 1401348594, system 3160, user 927
Disk I/O: reads 0, writes 0
Console I/O: reads 25, writes 67
Paging: faults 0
Network I/O: packets received 0, sent 0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

## 6. Chương trình echo

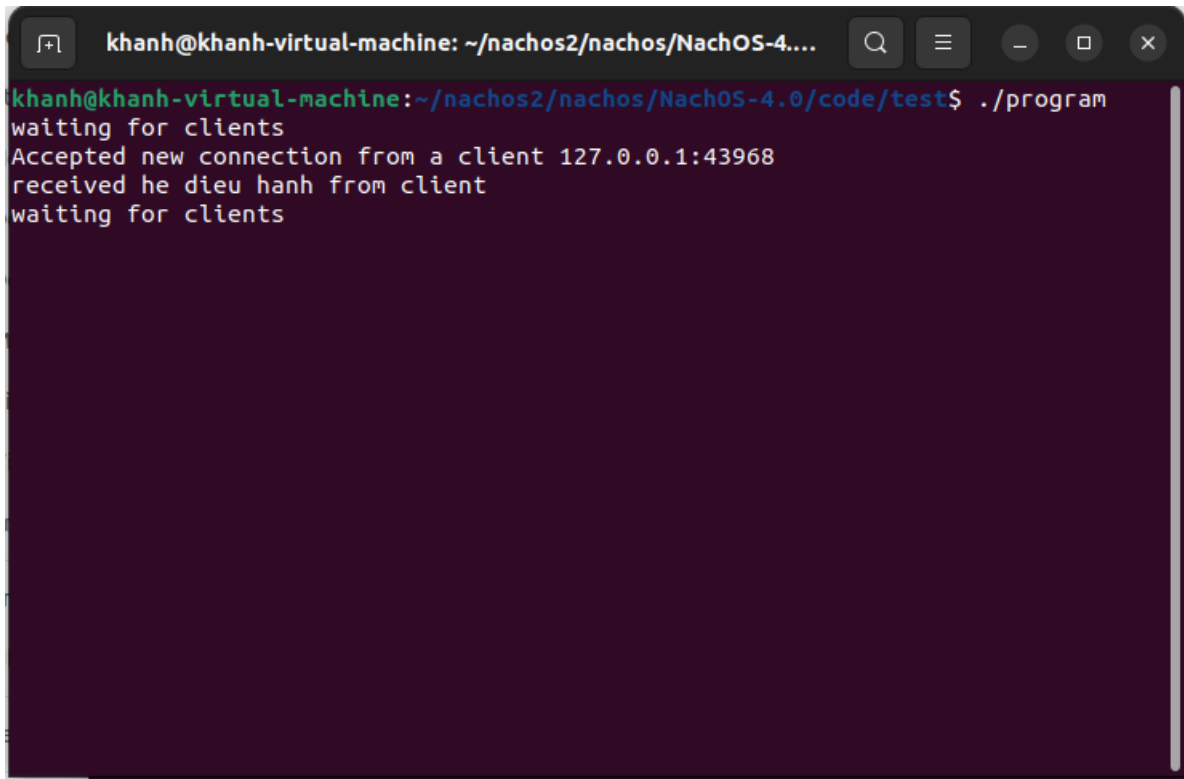
- Yêu cầu chương trình: viết chương trình kết nối socket tới server và gửi 1 tin nhắn, server nhận và chuyển đổi sang chữ hoa và gửi trả lại client.
- Cách thức hoạt động:
  - Bước 1: nhập chuỗi cần gửi.
  - Bước 2: Tạo 1 socket liên kết đến server.
  - Bước 3: Gửi chuỗi vừa nhập đến server.
  - Bước 4: Nhận chuỗi từ server và xuất lên màn hình.
- Hình ảnh demo chương trình:

Client:

A terminal window titled 'khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....'. The prompt is 'khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test\$'. The user enters './build.linux/nachos -x echo'. The program outputs 'Nhap chuoi gui di: he dieu hanh' and 'HE DIEU HANH'. It then prints 'Shutdown, initiated by user program.Machine halting!'. Finally, it displays system statistics: 'Ticks: total 615553096, idle 615551553, system 1470, user 73', 'Disk I/O: reads 0, writes 0', 'Console I/O: reads 13, writes 31', 'Paging: faults 0', and 'Network I/O: packets received 0, sent 0'. The prompt returns to 'khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test\$'.

Server:





```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ./program
waiting for clients
Accepted new connection from a client 127.0.0.1:43968
received he dieu hanh from client
waiting for clients
```

## 7. Chương trình filetransfer

- Yêu cầu chương trình: viết chương trình kết nối socket tới server và client đọc từ file lên gửi tin nhắn cho server, server nhận và chuyển đổi sang chữ hoa và gửi trả lại client. Client ghi lại xuống file.
- Cách thức hoạt động:
  - Bước 1: nhập tên file để Client đọc.
  - Bước 2: Tạo 1 socket liên kết đến server.
  - Bước 3: Gửi chuỗi vừa đọc đến server.
  - Bước 4: Nhận chuỗi từ server và ghi xuống file resultTran.txt.
- Hình ảnh demo chương trình:

Client:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x filetransfer
Nhap ten file ban muon doc: filedich.txt

Ghi file thanh cong

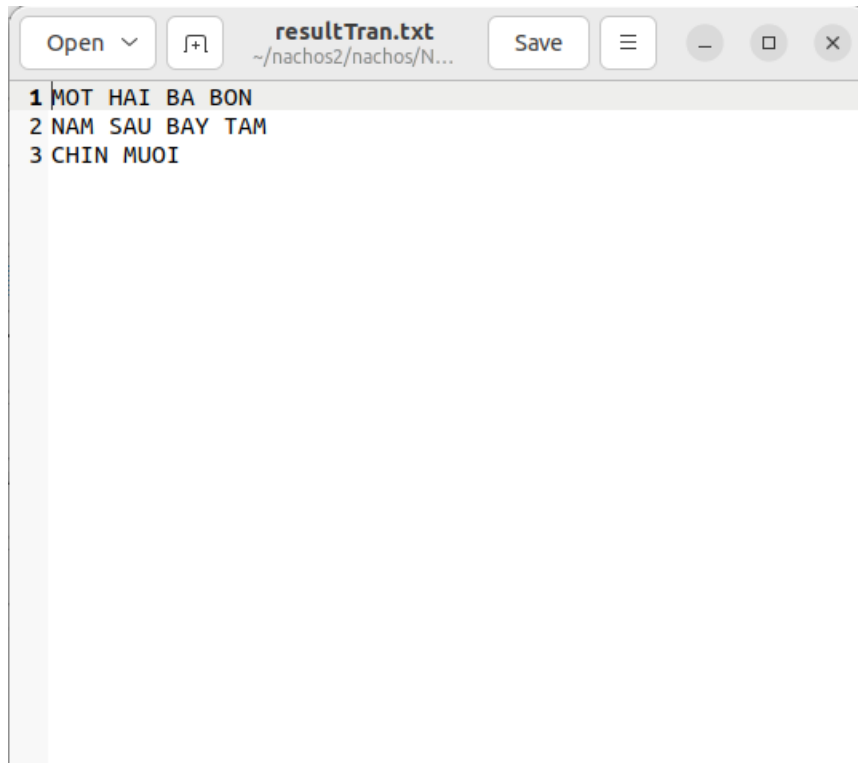
Shutdown, initiated by user program.Machine halting!

Ticks: total 805351240, idle 805348342, system 2150, user 748
Disk I/O: reads 0, writes 0
Console I/O: reads 13, writes 49
Paging: faults 0
Network I/O: packets received 0, sent 0
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$
```

Server:

```
khanh@khanh-virtual-machine: ~/nachos2/nachos/NachOS-4....
khanh@khanh-virtual-machine:~/nachos2/nachos/NachOS-4.0/code/test$ ./program
waiting for clients
Accepted new connection from a client 127.0.0.1:52478
received mot hai ba bon
nam sau bay tam
chin muoi
from client
waiting for clients
█
```

Nội dung file:



## **SPECIFIC PROBLEMS AND ISSUES**

- Lúc đầu hơi khó khăn khi sử dụng 2 hàm `System2User()` và `User2System()`.
- Không hiểu cách hoạt động của `FILESYSTEM_STUB` code. Nên cố fixbug khi các hàm thông báo không được định nghĩa.
- Chỉ sử dụng hàm `Read/ Write` thay cho `Send/ Receive`. Code xong chương trình không chạy được.
- Lúc đầu hơi khó khăn để hiểu và tìm kiếm các hàm thực thi trong các thư viện của hệ thống.