

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP HCM
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN ĐỒ ÁN 1
LẬP TRÌNH NACHOS

CÁC THÀNH VIÊN THAM GIA

1. Phan Gia Hân – 18120026
2. Đặng Văn Hiến – 18120363
3. Lê Thanh Viễn – 18120647

NĂM HỌC 2020-2021

Contents

1. THÔNG TIN THÀNH VIÊN	3
2. BẢNG PHÂN CÔNG CÔNG VIỆC:	3
3. CÀI ĐẶT TỔNG QUAN.....	3
4. CÀI ĐẶT SYSTEM CALL VÀ EXCEPTION	3
4.1 Cài đặt lại các exception:	3
4.2 Cài đặt hàm IncreasePC():	4
4.3 Thêm class synchcons :.....	4
4.4 Cài đặt các SyscallException :	5
4.5 Viết chương trình test các sycall:.....	1
4.6 Biên dịch mã nguồn NachOS:.....	2
5. DEMO HÌNH ẢNH MINH HỌA	2
5.1 Test creatfile.c :	2
5.2 Test echo.c :	3
5.3 Test cat.c :	3
5.4 Test copy.c :	4
6. TÀI LIỆU THAM KHẢO.....	4

1. THÔNG TIN THÀNH VIÊN

MSSV	Họ Tên	Email
18120026	Phan Gia Hân	18120026@student.hcmus.edu.vn
18120363	Đặng Văn Hiến	18120363@student.hcmus.edu.vn
18120647	Lê Thanh Viễn	18120647@student.hcmus.edu.vn

2. BẢNG PHÂN CÔNG CÔNG VIỆC:

Mức độ hoàn thành công việc:

Người thực hiện	Công việc	Mức độ hoàn thành
Phan Gia Hân	Câu 1, 2,3	100%
Đặng Văn Hiến	Câu 4,5,6,7	100%
Lê Thanh Viễn	Câu 8,9,10,11	100%

3. CÀI ĐẶT TỔNG QUAN

- Cài phần mềm hỗ trợ tạo máy ảo VMware Workstation.
- Tải hệ điều hành CentOS 6.
- Cài đặt CentOS và đăng nhập root user.
- Cài đặt gcc với yum – thực thi lệnh trên terminal :
 - o \$ yum -y install gcc
 - o \$ yum -y install gcc-c++
 - (* dùng lệnh \$ gcc --version để check lại)
- Chuyển mã nguồn NachOS (tải từ moodle) thông qua SharedFolder
- Copy thư mục nachos vào thư mục root/hdh :
 - o \$ yes | cp -rf /mnt/hgfs/ShareFolder/nachos ~/hdh

4. CÀI ĐẶT SYSTEM CALL VÀ EXCEPTION

4.1 Cài đặt lại các exception:

B1: Vào file code/machine/machine.h lấy danh sách các exception.

B2: Viết lại chương trình bắt tất cả các exception đã lấy ở B1 bằng cách: Mỗi exception chỉ việc hiện ra thông báo loại exception và gọi hàm Halt(). Sửa trong file code/userprog/exception.cc

Vd: case *PageFaultException*:

```
DEBUG('a', "\n No valid translation found");  
  
printf("\n\n No valid translation found");  
  
interrupt->Halt(); //Dung de tat he dieu hanh  
  
break;
```

4.2 Cài đặt hàm IncreasePC():

-Trong file code/userprog/exception.cc ta thêm hàm IncreasePC() dùng để làm tăng Programming Counter để nạp lệnh tiếp theo để thực hiện.

-Ta thực hiện IncreasePC() theo các bước

B1: Lấy giá trị PC hiện tại lưu vào biến counter (tự tạo).

B2: Gán giá trị của PC trước bằng counter.

B3: Lấy giá trị PC kế tiếp lưu vào biến counter.

B4: Gán giá trị của PC hiện tại bằng counter.

B5: Gán giá trị của PC kế tiếp = counter + 4 để tiếp tục nạp lệnh.

4.3 Thêm class synchcons :

-Copy synchcons.h và synchcons.cc vào thư mục code/threads/

-Mở file code/Makefile.common, thêm synchcons.* vào USERPROG_* (.h .cc .o)

```
USERPROG_H = ../userprog/addrspace.h\  
..\  
../threads/synchcons.h  
  
USERPROG_C = ../userprog/addrspace.cc\  
..\  
../threads/synchcons.cc  
  
USERPROG_O = addrspace.o bitmap.o exception.o progtest.o console.o machine.o \  
mipssim.o translate.o synchcons.o
```

-Khai báo biến toàn cục gSynchConsole trong threads/system.h

```
#ifndef USER_PROGRAM
#include "machine.h"
#include "synchcons.h"
extern Machine* machine;           // user program memory and registers
extern SynchConsole* gSynchConsole; // declare global gSynchConsole
#endif
```

Cấp phát biến gSynchConsole trong threads/system.cc

```
#ifndef USER_PROGRAM // requires either FILESYS or FILESYS_STUB
Machine *machine;    // user program memory and registers
SynchConsole* gSynchConsole;
```

```
#ifdef USER_PROGRAM
    machine = new Machine(debugUserProg); // this must come first
    gSynchConsole = new SynchConsole();    // allocate gSynchConsole
#endif
```

4.4 Cài đặt các SyscallException :

-Define loại syscall trong file
code/userprog/syscall.h

```
#define SC_Open      5
#define SC_Read      6
#define SC_Write     7
#define SC_Close     8
#define SC_Fork      9
#define SC_Yield    10
#define SC_Seek     11
#define SC_ReadConsole 12
#define SC_PrintConsole 13
// Define new Syscall
```

-Khai báo các hàm syscall trong file
code/userprog/syscall.h

```
OpenFileId Open(char *name, int type);

/* Write "size" bytes from "buffer" to the open file
void Write(char *buffer, int size, OpenFileId id);
```

```
int Seek(int pos, OpenFileId id);
int ReadConsole(char buffer[], int length);
int PrintConsole(char buffer[]);
```

```

case SyscallException:
{
    switch (type)
    {
        case SC_Halt:
        { ...
        case SC_Create:
        { ...
        case SC_Open:
        { ...
        case SC_Close:
        { ...
        case SC_Read:
        { ...
        case SC_Write:
        { ...
        case SC_Seek:
        { ...
        case SC_ReadConsole:
        { ...
        case SC_PrintConsole:
        { ...

    default:
        printf("\n Unexpected us
        interrupt->Halt();
    }
}

```

-Mở file code/userprog/exception.cc

-Cài đặt xử lí cho mỗi loại syscall với switch (type) trong case SyscallException.

-Các tham số nhận được thông qua các thanh ghi:

// tham số thứ 1 sẽ được đưa vào thanh ghi r4

// tham số thứ 2 sẽ được đưa vào thanh ghi r5

// tham số thứ 3 sẽ được đưa vào thanh ghi r6

// tham số thứ 4 sẽ được đưa vào thanh ghi r7

-Kết quả trả về thông qua thanh ghi r2

-Dữ liệu cung cấp nếu là vùng nhớ thì được copy vào System space (kernel mode) đảm bảo tách biệt khỏi User space , sau khi xử lí sẽ copy về User space. Việc này được thực hiện thông qua hàm User2System, System2User.

-Khi thực hiện đọc, ghi ra console ta sử dụng hàm gSynchConsole->Read và gSynchConsole->Write từ biến toàn cục gSynchConsole

*Các syscall thao tác với file được xây dựng trên các hàm của class FileSystem và OpenFile

- class FileSystem : (code/filesys/filesys.h)
 - Khai báo bảng file (giới hạn 10 OpenFile);
 - Thêm các hàm quản lý chỗ trống trong bảng FindFreeSlot()
 - Lưu trữ type OpenFile *Open(char *name, int type)
- class OpenFile : (code/filesys/openfile.h)
 - Thêm biến lưu trữ type của filestream
 - Thêm phương thức Seek, GetCurrentPos.

- **SC_Open** – Tham số nhận vào địa chỉ tên file và loại filestream

// 0: read and write

// 1: read only

// 2: stdin

// 3: stdout

Trả về id của file trong bảng OpenFile *[10]; nếu thất bại trả về -1

- **SC_Close** – Tham số nhận vào ID filestream cần đóng

Trả về 0 nếu thành công; nếu thất bại trả về -1

- **SC_Read, SC_Write** – Tham số nhận vào địa chỉ buffer, số byte cần đọc/ ghi, và ID filestream
Trả về số byte thực sự đọc/ghi được ; nếu thất bại trả về -1; trả về -2 nếu EOF
- **SC_Seek** – Tham số nhận vào vị trí con trỏ cần di chuyển đến và ID filestream
Trả về vị trí con trỏ nếu thành công ; nếu thất bại trả về -1
- **SC_ReadConsole, SC_PrintConsole** – Chức năng đọc/ghi trên console
Trả về số byte thực sự đọc/ghi được ;
Dựa trên phương thức class SynchConsole

4.5 Viết chương trình test các syscall:

-Mở thư mục code/test. Tạo các file source c : createfile.c, echo.c, cat.c, copy.c, viết các chương trình con dùng các syscall cần test thông qua include "syscall.h".

*Lưu ý: phải khai báo tất cả các biến ngay đầu block.

-Sử dụng các hàm cần test.

-Thêm vào file code/test/start.c và code/test/start.s phần thân hàm để gọi syscallexception tương ứng.

-Mở file code/test/Makefile :

+Thêm đường dẫn compiler, tùy chỉnh cross compiler:

```
GCCDIR = ../../../../gnu-decstation-ultrix/decstation-ultrix/2.95.3/
LD_FLAGS = -T script -N
AS_FLAGS = -mips2
CPP_FLAGS = $(INCDIR)
```

```
CC = $(GCCDIR)gcc -B../../../../gnu-decstation-ultrix/
AS = $(GCCDIR)as
LD = $(GCCDIR)ld
```

+ Thêm các file cần biên dịch với lệnh gmake all

```
all: halt shell matmult sort echo createfile copy cat
```

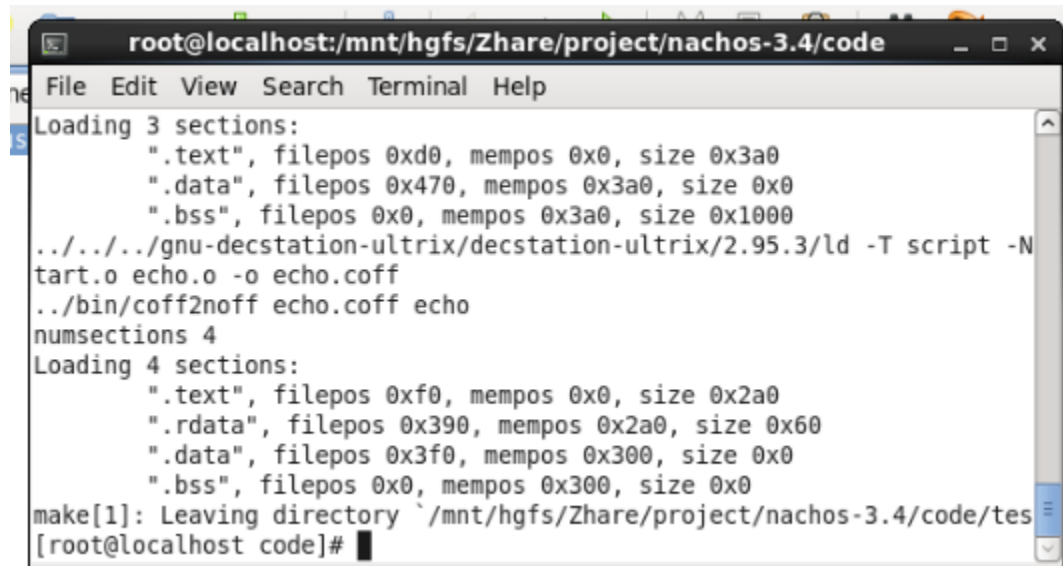
-Thêm thông tin biên dịch với mỗi file tương ứng:

```
copy.o: copy.c
$(CC) $(CFLAGS) -c copy.c
copy: copy.o start.o
$(LD) $(LD_FLAGS) start.o copy.o -o copy.coff
../bin/coff2noff copy.coff copy
```

```
.globl Seek
.ent Seek
Seek:
    addiu $2,$0,SC_Seek
    syscall
    j $31
.end Seek
```

4.6 Biên dịch mã nguồn NachOS:

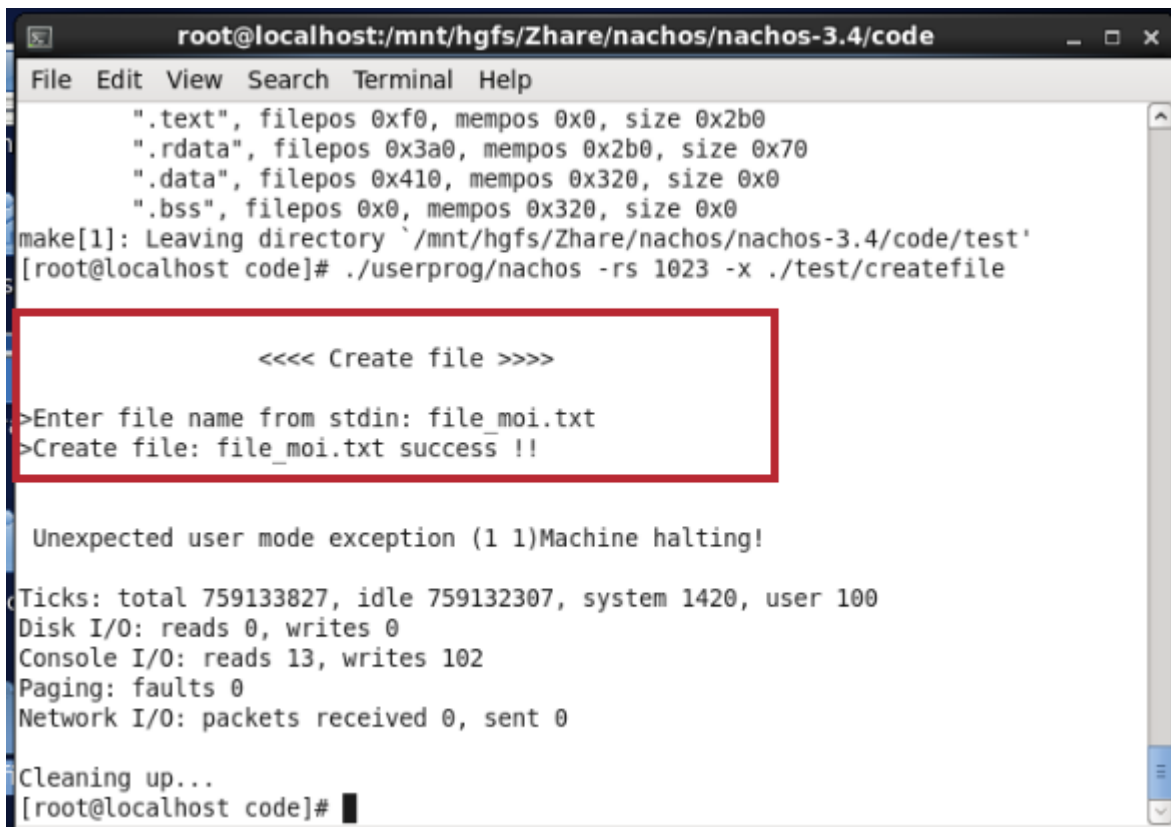
Tại thư mục code, ta mở Terminal (root user) thực thi lệnh gmake all.



```
root@localhost:/mnt/hgfs/Zhare/project/nachos-3.4/code
File Edit View Search Terminal Help
Loading 3 sections:
  ".text", filepos 0xd0, mempos 0x0, size 0x3a0
  ".data", filepos 0x470, mempos 0x3a0, size 0x0
  ".bss", filepos 0x0, mempos 0x3a0, size 0x1000
../../../../gnu-decstation-ultrix/decstation-ultrix/2.95.3/ld -T script -N
tart.o echo.o -o echo.coff
../bin/coff2noff echo.coff echo
numsections 4
Loading 4 sections:
  ".text", filepos 0xf0, mempos 0x0, size 0x2a0
  ".rdata", filepos 0x390, mempos 0x2a0, size 0x60
  ".data", filepos 0x3f0, mempos 0x300, size 0x0
  ".bss", filepos 0x0, mempos 0x300, size 0x0
make[1]: Leaving directory `/mnt/hgfs/Zhare/project/nachos-3.4/code/test'
[root@localhost code]#
```

5. DEMO HÌNH ẢNH MINH HỌA

5.1 Test creatfile.c : [root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/createfile



```
root@localhost:/mnt/hgfs/Zhare/nachos/nachos-3.4/code
File Edit View Search Terminal Help
  ".text", filepos 0xf0, mempos 0x0, size 0x2b0
  ".rdata", filepos 0x3a0, mempos 0x2b0, size 0x70
  ".data", filepos 0x410, mempos 0x320, size 0x0
  ".bss", filepos 0x0, mempos 0x320, size 0x0
make[1]: Leaving directory `/mnt/hgfs/Zhare/nachos/nachos-3.4/code/test'
[root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/createfile

<<<< Create file >>>>

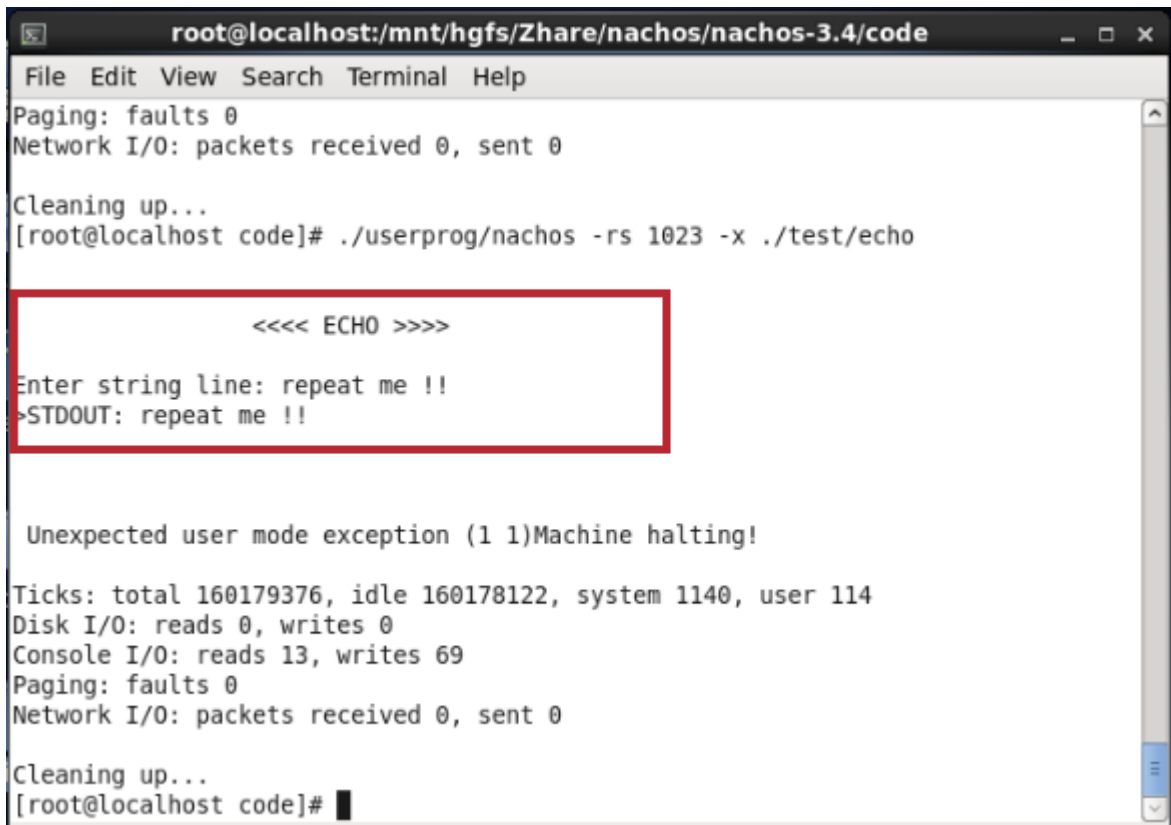
>Enter file name from stdin: file_moi.txt
>Create file: file_moi.txt success !!

Unexpected user mode exception (1 1)Machine halting!

Ticks: total 759133827, idle 759132307, system 1420, user 100
Disk I/O: reads 0, writes 0
Console I/O: reads 13, writes 102
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[root@localhost code]#
```


5.2 Test echo.c : `[root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/echo`



```
root@localhost:/mnt/hgfs/Zhare/nachos/nachos-3.4/code
File Edit View Search Terminal Help
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/echo

<<<< ECHO >>>>

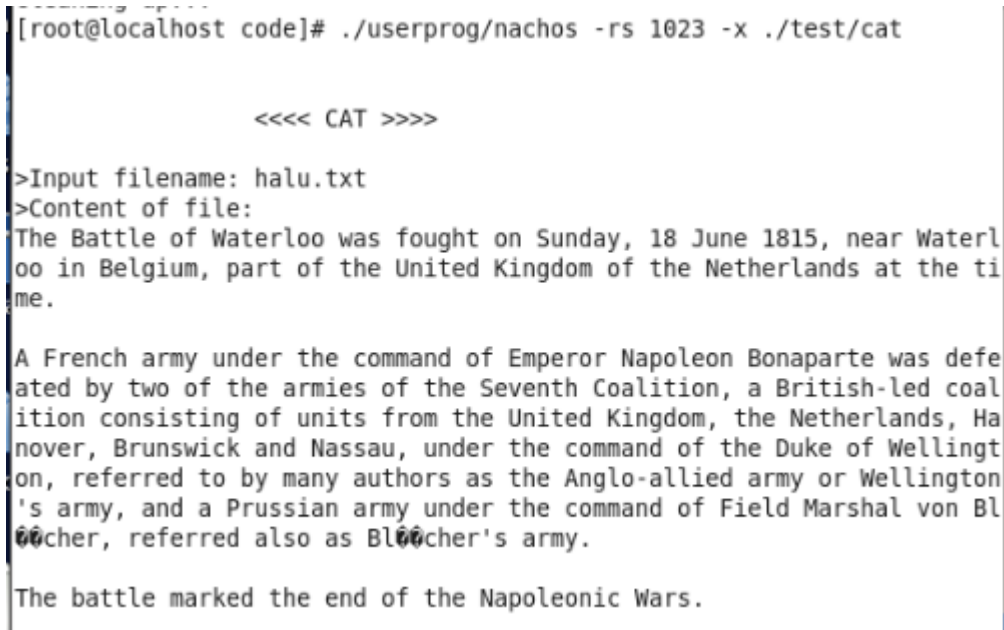
Enter string line: repeat me !!
>STDOUT: repeat me !!

Unexpected user mode exception (1 1)Machine halting!

Ticks: total 160179376, idle 160178122, system 1140, user 114
Disk I/O: reads 0, writes 0
Console I/O: reads 13, writes 69
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[root@localhost code]#
```

5.3 Test cat.c :



```
[root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/cat

<<<< CAT >>>>

>Input filename: halu.txt
>Content of file:
The Battle of Waterloo was fought on Sunday, 18 June 1815, near Waterl
oo in Belgium, part of the United Kingdom of the Netherlands at the ti
me.

A French army under the command of Emperor Napoleon Bonaparte was defe
ated by two of the armies of the Seventh Coalition, a British-led coal
ition consisting of units from the United Kingdom, the Netherlands, Ha
nover, Brunswick and Nassau, under the command of the Duke of Wellingt
on, referred to by many authors as the Anglo-allied army or Wellington
's army, and a Prussian army under the command of Field Marshal von Bl
öcher, referred also as Blöcher's army.

The battle marked the end of the Napoleonic Wars.
```

5.4 Test copy.c :

```
[root@localhost code]# ./userprog/nachos -rs 1023 -x ./test/copy

<<<< COPY >>>>

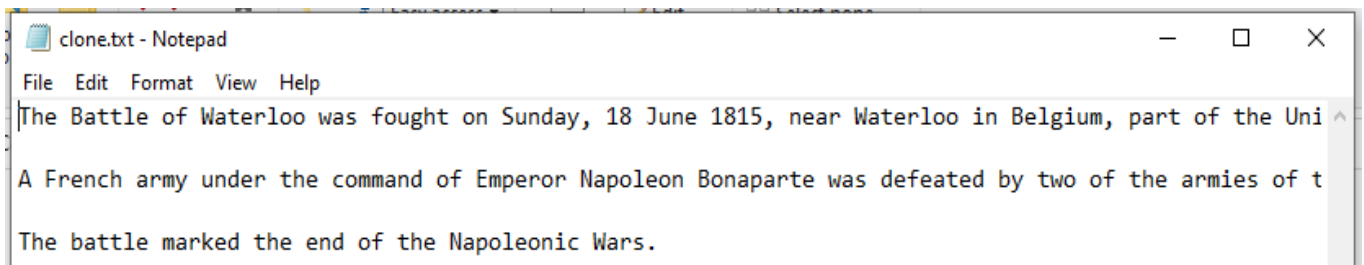
>Input source filename: halu.txt
>Input destination filename: clone.txt
>Copied successfully !!

Unexpected user mode exception (1 1)Machine halting!

Ticks: total 1361908781, idle 1361884154, system 3920, user 20707
Disk I/O: reads 0, writes 0
Console I/O: reads 19, writes 104
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
[root@localhost code]#
```

1 KB	clone.txt
0 KB	file_moi.txt
1 KB	halu.txt



clone.txt - Notepad

File Edit Format View Help

The Battle of Waterloo was fought on Sunday, 18 June 1815, near Waterloo in Belgium, part of the Uni...

A French army under the command of Emperor Napoleon Bonaparte was defeated by two of the armies of t...

The battle marked the end of the Napoleonic Wars.

6. TÀI LIỆU THAM KHẢO

- Các file hướng dẫn trên moodle