**HỆ ĐIỀU HÀNH**

**ĐỒ ÁN 01: NACHOS**

C:\Users\tdqua_000\Dropbox\SS-Slides\DeCuong-CDIO\Template CDIO v4.2\Templates\Hinh anh\LogoTruong.png

Bộ môn Mạng máy tính

Khoa Công nghệ thông tin

Đại học Khoa học tự nhiên TP HCM

**MỤC LỤC**

[I. Thông tin nhóm 3](#_Toc447745973)

[II. Nội dung giải quyết 4](#_Toc447745974)

[1. Về các exception 4](#_Toc447745975)

[2. Nhận systemcall 4](#_Toc447745976)

[3. Increase PC 4](#_Toc447745977)

[4. Chèn lớp SynchConsole vào trở thành biến toàn cục 4](#_Toc447745978)

[5. Viết systemcall ReadInt 4](#_Toc447745979)

[6. Viết systemcall PrintInt 5](#_Toc447745980)

[7. Viết systemcall ReadChar 5](#_Toc447745981)

[8. Viết systemcall PrintChar 6](#_Toc447745982)

[9. Viết systemcall ReadString 6](#_Toc447745983)

[10. Viết systemcall PrintString 6](#_Toc447745984)

[11. Viết chương trình help 7](#_Toc447745985)

[12. Viết chương trình ascii 7](#_Toc447745986)

[13. Viết chương trình sort 7](#_Toc447745987)

# Thông tin nhóm

|  |  |
| --- | --- |
| **MSSV** | **Họ Tên** |
| 1512151 | Lâm Khả Hân |
| 1512172 | Phạm Thị Thanh Hoài |
| 1512581 | Đào Minh Toàn |

# Nội dung giải quyết

## 1. Về các exception

Chúng em làm một hàm if else với các điều kiện xét cụ thể cho 2 trường hợp đặc biệt là NoException và SyscallException, còn lại chúng em báo lỗi. Với SyscallException, chúng em làm kỹ hơn với việc đưa vào sử dụng một lệnh switch case để phân nhóm xử lý System call sau này.

Còn với NoException, chúng em trả về một dòng Debug thôi ạ.

## 2. Nhận systemcall

System call trong đồ án lần này em không thấy có nhiều yêu cầu mà chỉ yêu cầu viết lại chương trình để switch xử lý SC\_Halt. Chúng Chúng em đã dùng switch case để xử lý và cũng để cho những câu hỏi khác ạ.

## 3. Increase PC

Chúng Chúng em đã copy và viết lại một tí cho tối ưu, tên hàm của nó là increasePC(). Việc thêm vào phần sử lý systemcall cũng khá đơn giản, chỉ việc thêm trường default và thêm dòng đó vào cùng với lệnh break;

## 4. Chèn lớp SynchConsole vào trở thành biến toàn cục

Sau quá trình tham khảo trong bài liệu hướng dẫn thực hành, Chúng em đã thêm vào các file threads/system.cc và threads/system.h các đoạn code phù hợp cho thư viện SynchConsole. Cụ thể là ở những đoạn có #ifdef USER\_PROGRAM

## 5. Viết systemcall ReadInt

Chúng em đã thêm một case là SC\_ReadInt đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 11.

Ý tưởng của nhóm chúng em là cho nó lưu vào con trò buf với max length là 100 (99 ký tự tối đa – macro BUFFER\_LENGTH). Sau đó, dùng gSynchConsole như ở trên để đọc chuỗi với max buffer là BUFFER\_LENGTH – 1. Cuối cùng cho nó qua hàm ConvertStringToInteger để chuyển nó thành số với điều kiện trong đó chỉ là chuỗi số, nếu tồn tại một ký tự không phải số thì sẽ trả về 0, còn không sẽ trả về atoi của chuỗi số đó!

Bọn Chúng em đã xử lý kỹ, chấp người dùng phá cũng không thể sụp hệ điều hành ở system call này.

Kết quả, bọn em trả về qua thanh ghi số 2.

Chúng em có làm một test để kiểm thử tên là readint.c (có kết hợp SC\_PrintInt)

## 6. Viết systemcall PrintInt

Chúng em đã thêm một case là SC\_PrintInt đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 12.

Ý tưởng của nhóm chúng em là dùng hàm sprint để chuyển số thành chuỗi vào biến buf, sau đó dùng gSynchConsole để xuất ra monitor thông qua phương thức Write.

Bọn Chúng em đã xử lý kỹ, chấp người dùng phá cũng không thể sụp hệ điều hành ở system call này.

Bọn em truyền tham số qua thanh ghi số 4.

Chúng em có làm một test để kiểm thử tên là printint.c

## 7. Viết systemcall ReadChar

Chúng em đã thêm một case là SC\_ReadChar đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 13.

Ý tưởng của nhóm chúng em là cho nó lưu vào con trò buf với max length là 100 (99 ký tự tối đa – macro BUFFER\_LENGTH). Bọn em biết rằng, hoàn toàn có thể đọc một ký tự bằng phương thức Read với 1 ký tự. Nhưng không, chúng em muốn vận dụng một tí để kiểm thử hệ điều hành này. Và chúng Chúng em đã dùng một chuỗi static với buffer length bằng BUFFER\_LENGTH để lưu lại cho lần sau xử lý  khi người dùng nhập vô một chuỗi dài thì kernel sẽ lần lượt đọc cho hết và sẽ chỉ trả về những ký tự đầu tiên rồi thứ 2,… thứ n cho đến khi hết string thì người dùng mới nhập tiếp (hoăc dùng ReadString).

Bọn Chúng em đã xử lý kỹ, chấp người dùng phá cũng không thể sụp hệ điều hành ở system call này.

Kết quả, bọn em trả về qua thanh ghi số 2.

## 8. Viết systemcall PrintChar

Chúng em đã thêm một case là SC\_PrintChar đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 14.

Cái này đơn giản hơn, bọn em chỉ việc sprint ra buf với ký tự dược truyền qua thanh ghi số 4 là xong.

Sau đó dùng gSynchConsole xuất ra, thế là xong.

## 9. Viết systemcall ReadString

Chúng em đã thêm một case là SC\_ReadString đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 15.

Chúng em dùng 2 thanh ghi 4, 5 lần lượt để truyền 2 tham số char buf[] và int len vào với ý nghĩa như đề bài yêu cầu. Khi test, chúng em để BUFFER\_LENGTH lại còn 5 (tức là đọc mỗi lần tối đa 4 ký tự từ console), và ở user space cho chương trình đọc một chuỗi tối đa 20 ký tự (21 buffer length).

Ý tưởng: chúng em sẽ dùng một biến đếm để đo số lượng char đã được nạp vào bộ nhớ userspace và một biến khác để xác định sẽ đọc bao nhiêu ký tự ở lượt xử lý này. Sau đó cho nó lặp while do và đẩy vào user space qua phương thức System2User. Quá đơn giản.

Ví dụ: nhập chuỗi “Khanh dep trai hon Dat”. Kernel sẽ xử lý lần lượt như sau

|  |  |  |
| --- | --- | --- |
| STT | Kernel | User |
| 1 | Khan ((strlen) 4 == 4 (BUFFER\_LENTH – 1)  continue) | Khan |
| 2 | h de (4 == 4  continue) | Khanh de |
| 3 | p tr (4 == 4  continue) | Khanh dep tr |
| 4 | ai h (4 == 4  continue) | Khanh dep trai h |
| 5 | on D (4 == 4  continue) | Khanh dep trai hon D |
| 6 | At (2 < 4  break) | Khanh dep trai hon Dat |

## 10. Viết systemcall PrintString

Chúng em đã thêm một case là SC\_PrintString đồng thời với định nghĩa ID của nó trong file userprog/syscall.h là 16.

Cũng không quá khó, vì hầu như thư viện có hết rồi, chúng em cũng chả làm gì nhiều cả. Chỉ việc dùng phương thức User2System đọc vào buffer với mức BUFFER\_LENGTH – 1 cho trước, đọc chừng nào không đọc được nữa (strlen(buf) == 0) thì thôi. Cứ mỗi lần đọc như vậy, ta lại gửi ra monitor cho User qua thư viện gSynchConsole.

Ví dụ: in chuỗi “Khanh dep trai hon Dat”. Kernel sẽ xử lý lần lượt như sau

|  |  |  |
| --- | --- | --- |
| STT | Kernel | Monitor |
| 1 | Khan (4 > 0  continue) | Khan |
| 2 | h de (4 > 0  continue) | Khanh de |
| 3 | p tr (4 > 0  continue) | Khanh dep tr |
| 4 | ai h (4 > 0  continue) | Khanh dep trai h |
| 5 | on D (4 > 0  continue) | Khanh dep trai hon D |
| 6 | At (2 > 0  continue) | Khanh dep trai hon Dat |
| 7 | (0 > 0  false  break) |  |

## 11. Viết chương trình help

Chúng em sử dụng các hàm đã được tạo lập sẵn trong system calls để thực hiện hàm help.

Trong đó các hàm chính cần phải sử dụng chí có mỗi hàm PrintString() vì chỉ có in ra thông tin nhóm và các chức năng thêm của hàm ASCII() và hàm Sort().

Ngoài ra khi in ra các nội dung thông tin chúng em phải phân nội dung ra nhiều lần PrintString() khác nhau vì nếu để một chung 1 luồng như vậy khi gmake all máy tính sẽ thông báo eror.

## 12. Viết chương trình ascii

Hàm ASCII() viết rất đơn giản vì do đã có hàm chuyển từ số về sang ASCII nên việc đơn giản chỉ cần tạo lập dòng for cùng các điều kiện cần hoạc dùng vòng lặp while tùy sở thích mỗi người sau đó tạo lập dấu {} bên trong {} gọi hàm PrintChar() để trả số về lại mã ASCII còn dấu \n hay \t thêm vào nhìn cho có giãn cách đẹp mắt thị gọi thêm hàm PrintString().

## 13. Viết chương trình sort

Ý tưởng chính của hàm sort này là thuật toán Buble Sort (Sắp Xếp Nổi Bọt)

Giới hạn phần tử của mảng không quá 100.Ở đây chúng em dùng mãng tĩnh cho dễ sử dụng và không phải cấp phát lung tung.Chúng em có đặt điều kiện để khi người dùng nhập phần tử sai ví dụ như nhỏ hơn 0 hoặc lớn hơn 100 thì sẽ phải nhập lại.Tiếp theo đó là vòng lặp để đưa số vào các phần tử bên trong mảng cuối cùng là sort và in ra kết quả lại trên console.