|  |
| --- |
| Họ và tên: Nguyễn Trần Bảo Anh  Mã số sinh viên: 22520066  Lớp: IT007.O21.CNVN.1 |

HỆ ĐIỀU HÀNH  
BÁO CÁO LAB 6

**CHECKLIST**

**6.4. BÀI TẬP THỰC HÀNH**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Câu 1** | **Câu 2** | **Câu 3** | **Câu 4** | **Câu 5** |
| **Trình bày giải thuật** |  |  |  |  |  |
| **Chụp hình minh chứng (chạy ít nhất 3 lệnh)** |  |  |  |  |  |
| **Giải thích code, kết quả** |  |  |  |  |  |

**Tự chấm điểm:** 9

*\*Lưu ý: Xuất báo cáo theo định dạng PDF, đặt tên theo cú pháp:* ***<Tên nhóm>\_LAB6.pdf***

**Source code**

**A screen shot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A computer screen shot of a program

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A computer screen shot of text

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**

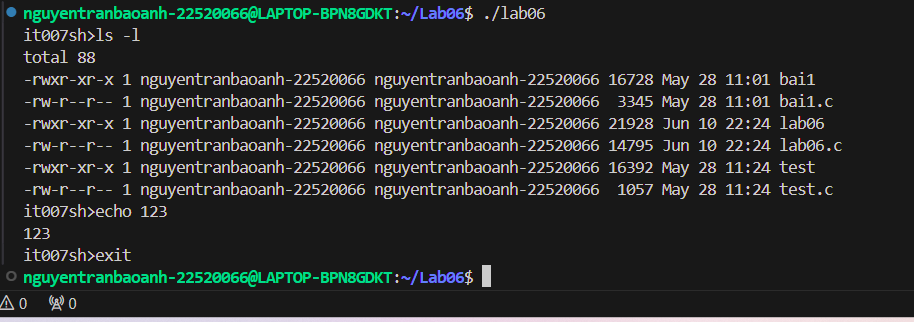
**A screen shot of a computer program

Description automatically generated**

**6.4. BÀI TẬP THỰC HÀNH**

# Câu 1

Khi muốn thoát chương trình thì gõ lệnh “exit”



Khi lệnh thực thi không đúng sẽ báo lỗi

A black screen with green text

Description automatically generated

**Giải thích kết quả:**

- Với lệnh cat abc.txt, trong file abc.txt có dòng text: Nguyen Thi Ngoc Tram. Ban

đầu, chương trình sẽ copy lệnh trong trong history\_command (mảng lưu trữ lịch

sử lệnh) và kiểm tra lệnh có ký tự pipe hay không. Tất nhiên đây là lệnh đơn giản,

không có ký tự pipe nên giá trị iPipeExe ở đây gán là 0.

- Tiếp đến chương trình cấp phát bộ nhớ và tách lệnh thành các đối số riêng biệt

thông qua hàm Tokernizer.

- Tiếp theo, bước đầu ta kiểm tra lệnh có chứa ‘exit’ hay không. Nếu có thì sẽ kết

thúc chương trình. Tiếp đến sẽ kiểm tra lệnh có chứa ký tự ‘~’ hay cd. Nếu có

chương trình sẽ trả về đường dẫn Home hoặc đường dẫn đầy đủ của thư mục

tương ứng với các lệnh.

- Nếu không phải các lệnh trên, chương trình sẽ tạo ra tiến trình con thực thi lệnh

thông qua lệnh execvp(). Nếu thành công ret được gán sẽ có giá trị là 1 và ngược

lại lệnh sẽ không thực thi được và ret trả về -1 – thông báo lỗi. Ở đây, tiến trình

con sẽ thực hiện lệnh cat abc.txt và in ra dòng text của file: Nguyen Thi Ngoc

Tram như trên màn hình.

- Cuối cùng, tiến trình cha sẽ đợi tiến trình con kết thúc và trả các giá trị cờ hiệu về

giá trị ban đầu, giải phóng bộ nhớ và kết thúc, in ra it007> chờ người dùng nhập

lệnh tiếp theo. Nếu lệnh tiếp theo là exit thì sẽ kết thúc luôn chương trình.

- Tương tự với các lệnh tiếp theo.

# Câu 2

A computer screen shot of a computer code

Description automatically generated

**Giải thích kết quả:**

- Ban đầu, ta thực hiện câu lệnh cat abc.txt, chương trình sẽ lưu lệnh này trongmảng history\_command và biến đếm số lệnh lúc này Count\_HF sẽ là 1.

- Tiếp đến ta thực hiện lệnh thứ 2 echo abc, chương trình sẽ lưu lệnh này trongmảng history\_command và biến đếm số lệnh lúc này Count\_HF sẽ là 2.

- Kế đến ta thực hiện lệnh thứ 3 là ls, chương trình sẽ lưu lệnh này trong mảnghistory\_command và biến đếm số lệnh lúc này Count\_HF sẽ là 3.

- Cuối cùng ta nhập vào bàn phím HF, chương trình kiểm tra và thấy chuỗi trên đang yêu cầu lịch sử câu lệnh. Kết quả chương trình in ra các câu lệnh đã sử dụngvà thực hiện lại câu lệnh gần nhất là lệnh ls. Do đó ta có thể thấy kết quả in ra màn hình là 3 câu lệnh đã được sử dụng trước đó và danh sách các thư mục thông qua lệnh ls.

# Câu 3

**Test 1:** ls > test1.txt

A screen shot of a computer

Description automatically generated

**Giải thích kết quả:**

- Với lệnh ls > test1.txt, ta thấy ban đầu file test1.txt là file trống. Khi ta thực hiện

lệnh này, trước tiên chương trình sẽ kiểm tra có ký tự pipe (“|”) hay không? Ở

trường hợp này sẽ là không và sẽ gán cho iPipeExe là 0 và không thực hiện lệnh

chứa pipe.

- Khi đó, ta sẽ cấp phát bộ nhớ cho mảng chứa tập lệnh, copy vào mảng chứa lịch sử

tập lệnh và tách chúng thành các đối số riêng biệt thông qua hàm Tokernizer và

lưu lại. Khi đó số lượng đối số của lệnh trên là 3 (ls, test1.txt, >) và chương trình

sẽ kiểm tra trong đối số này có ký tự “>” nên sẽ thực hiện hàm redirect\_output().

- Trong hàm redirect\_output(), trước khi thực hiện lệnh ls in ra các thư mục, chương

trình sẽ tạo 1 bản sao của file stdout là SaveStdout. Sau đó, chương trình sẽ mở

file test1.txt với các quyền tạo, chỉnh sửa. Và sẽ sao chép tất cả các thư mục trong

lệnh ls thực thi vào test1.txt (sau khi đã thực hiện hàm này) và đánh dấu cờ hiệu

iRedirectout đã thực hiện lên 1.

- Do đó, ta sẽ không thấy kết quả in ra màn hình mà các kết quả đã được chuyển

hướng vào file test1.txt. Để kiểm tra, ta mở file test1.txt và thấy các thư mục đã

được lưu vào đó.

**Test 2:** sort < test2.txt

**A computer screen shot of a black screen

Description automatically generated**

**Giải thích kết quả:**

- Với lệnh sort < test2.txt, ta thấy ban đầu file test2.txt có các chữ số sắp xếp không

theo thứ tự: 6,3,1,5,9,8,0. . Khi ta thực hiện lệnh này, trước tiên chương trình sẽ

kiểm tra có ký tự pipe (“|”) hay không? Ở trường hợp này sẽ là không và sẽ gán

cho iPipeExe là 0 và không thực hiện lệnh chứa pipe.

- Khi đó, ta sẽ cấp phát bộ nhớ cho mảng chứa tập lệnh, copy vào mảng chứa lịch sử

tập lệnh và tách chúng thành các đối số riêng biệt thông qua hàm Tokernizer và

lưu lại. Khi đó số lượng đối số của lệnh trên là 3 (sort, test2.txt, <) và chương trình

sẽ kiểm tra trong đối số này có ký tự “<” nên sẽ thực hiện hàm redirect\_input().

- Trong hàm redirect\_input(), trước khi thực hiện lệnh sort để sắp xếp các phần tử

trong thư mục test2.txt, chương trình sẽ tạo 1 bản sao của file stdin là SaveStdin.

Sau đó, chương trình sẽ mở file test2.txt với quyền đọc. Sau đó sẽ thực hiện lệnh

dup2() để sao chép file sao chép file descriptor vào stdin. Có nghĩa là bất cứ đầu

đọc nào từ stdin đều sẽ chuyển hướng sang file này. Và đánh dấu cờ hiệu

iRedirectin đã thực hiện lên 1.

- Điều này có nghĩa ta thực hiện hàm sort với đầu đọc được chuyển hướng đến file

trên chứa các phần tử trong file test2.txt. Do đó kết quả trên màn hình ta có thể

thấy các phần tử trong test2.txt đều đã được sắp xếp theo thứ tự từ bé đến lớn:

0,1,3,5,6,8,9.

# Câu 4

**Test 1:**

A screenshot of a computer

Description automatically generated

**Test 2:**

**A computer screen shot of a black screen

Description automatically generated**

**Test 3:**

**A computer screen shot of a black screen

Description automatically generated**

**Giải thích kết quả:**

- Như ở Testcase 1, với lệnh cat out.txt | sort, đầu tiên chương trình sẽ kiểm tra lệnh

này có ký tự pipe (“|”) hay không thông qua hàm ExecuteString,

parseCommandLine, Find\_pile\_char. Ở đây, hàm này sẽ trả giá trị về 1 rồi gán cho

iPipeExe và sẽ lưu vào parse ở phần thứ nhất và parsepipe ở phần thứ 2.

- Tiếp đến, chương trình sẽ kiểm tra giá trị iPipeExe ở trên và thực hiện hàm

Execute\_pipe.

- Ở hàm này, đầu tiên sẽ tạo ra một tiến trình con p1. Khi tiến trình con p1 thực thi

sẽ tạo ra 1 tiến trình con p2 và 1 pipe. Khi tiến trình p2 thực thi sẽ đóng đầu đọc

của pipe và đầu ghi của pipe được sao chép vào stdout thông qua hàm dup2(). Khi

đó lệnh cat out.txt sẽ được thực thi.

- Tiếp đến, khi tiến trình con p2 kết thúc, khi đó đầu ghi của pipe sẽ đóng lại và đầu

đọc lại được mở và sao chép vào stdin thông qua hàm dup2(). Lúc này, lệnh sort sẽ

thực hiện sắp xếp các phần tử có trong out.txt đã được đọc thông qua pipe rồi cuối

cùng in ra màn hình kết quả các phần tử đã được sắp xếp tăng dần (ở đây theo chữ

cái đầu, nếu trùng sẽ xét đến chữ cái kế tiếp).

# Câu 5

**A screenshot of a computer

Description automatically generated**

**Giải thích kết quả:**

- Đầu tiên, khi nhập Ctrl+C, terminal sẽ gửi một tín hiệu ngắt đến tiến trình hiện tại đang chạy.

- Khi tiến trình nhận được tín hiệu ngắt, tín hiệu này sẽ được xử lý thông qua hàm control\_sig với số hiệu signal(ở đây là 2 khi nhập Ctrl+C). Khi đó, hàm này sẽ dừng tiến trình hiện tại, xả bộ đệm và in ra một dòng it007> mới và tiếp tục chạy.