|  |
| --- |
| **ĐẠI HỌC QUỐC GIA TP. HCM**  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**    *BÁO CÁO BÀI THỰC HÀNH*  **HỆ ĐIỀU HÀNH**  Lớp: **IT007.N21.HTCL**    **LAB 3: TIẾN TRÌNH VÀ TIỂU TRÌNH**  ***Giảng viên:*** *ThS.**Lê Hoài Nghĩa*  ***Nhóm sinh viên thực hiện:***   * Nguyễn Thế Hưng – 21522120 – Nhóm trưởng * Nguyễn Lê Quỳnh Hương – 21520255 * Lê Nguyễn Nhật Anh – 21520138 * Nguyễn Minh Lý – 21521108   *Thành phố Hồ Chí Minh – Tháng 04/2023* |

**BẢNG PHÂN CÔNG CÔNG VIỆC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ và tên** | **Phụ trách** | **Trạng thái** |
| 1 | 21522120 | Nguyễn Thế Hưng | Bài 2 | Hoàn thành |
| 2 | 21520255 | Nguyễn Lê Quỳnh Hương | Bài 1 | Hoàn thành |
| 3 | 21520138 | Lê Nguyễn Nhật Anh | Bài 3 | Hoàn thành |
| 4 | 21521108 | Nguyễn Minh Lý | Bài 4 | Hoàn thành |

**3.5 Bài tập ôn tập**

**1. Mối quan hệ cha-con giữa các tiến trình**

**a. Vẽ cây quan hệ parent-child của các tiến trình bên dưới:**

UID PID PPID COMMAND

88 86 1 WindowServer

501 281 86 iTunes

501 282 86 Terminal

0 287 282 login

501 461 293 firefox-bin

501 531 86 Safari

501 726 86 Mail

501 751 293 Aquamacs

501 293 287 -bash

UID 88  
PID 86   
Window Sever

UID 501  
PID 281   
iTunes

UID 501  
PID 282   
Terminal

UID 501  
PID 531   
Safari

UID 501  
PID 726   
Mall

UID 0  
PID 287   
login

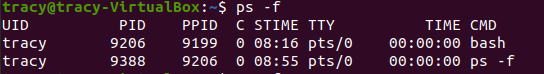
UID 501  
PID 293   
-bash

UID 501  
PID 461   
firefox bin

UID 501  
PID 751   
Aquamacs

**b. Trình bày cách sử dụng lệnh ps để tìm tiến trình cha của một một tiến trình dựa vào PID của nó.**

* Lệnh ps dùng để hiển thị chi tiết của một tiến trình. Khi sử dụng lệnh ps thì thông tin hiển thị bao gồm PID ( ID của tiến trình ) và PPID ( PID của tiến trình cha ).
* Sử dụng lệnh ***ps -f*** để tìm tiến trình cha của 1 tiến trình.



* Khi thực hiện lệnh ***ps -f***, ta có thể thấy **tiến trình bash có PID là 9202** là tiến trình cha của tiến trình ps -f thông qua PID và PPID mà khi dùng lệnh ps -f đã thể hiện.
* Ngoài ra khi biết được ID của một tiến trình, ta có thể sử dụng lệnh ***ps -fd [pidid]*** từ đó, có thể tìm PPID của tiến trình đó.
* Ví dụ minh họa sử dụng lệnh ***ps -fd 9206,*** ta có thể thấy PPID của tiến trình này là 9199.

Text

Description automatically generated

**c. Tìm hiểu và cài đặt lệnh pstree (nếu chưa được cài đặt), sau đó trình bày cách sử dụng lệnh này để tìm tiến trình cha của một tiến trình dựa vào PID của nó.**

* Lệnh pstree giúp hiển thị các tiến trình dưới dạng sơ đồ cây.
* Sử dụng lệnh ***pstree -sg [pidid]***, khi đó tiến trình cha của tiến trình này sẽ được thể hiện theo sơ đồ dạng cây.



* Ví dụ minh họa khi thực hiện lệnh ***pstree -sg 9206***, kết quả hiện thị tiến trình bash có PID = 9206 có tiến trình cha là tiến trình gnome-terminal có PID = 9199.

**2. Chương trình bên dưới in ra kết quả gì? Giải thích tại sao?**

/\*######################################

# University of Information Technology #

# IT007 Operating System #

# <Youe name>,<your Student ID> #

# File: exercise\_2.c #

######################################\*/

#include<stdio.h>

int main(){

pid\_t pid;

int num\_coconuts = 17;

pid = fork();

if(pid == 0) {

num\_coconuts = 42;

exit(0);

} else {

wait(NULL); /\*wait until the child terminates \*/

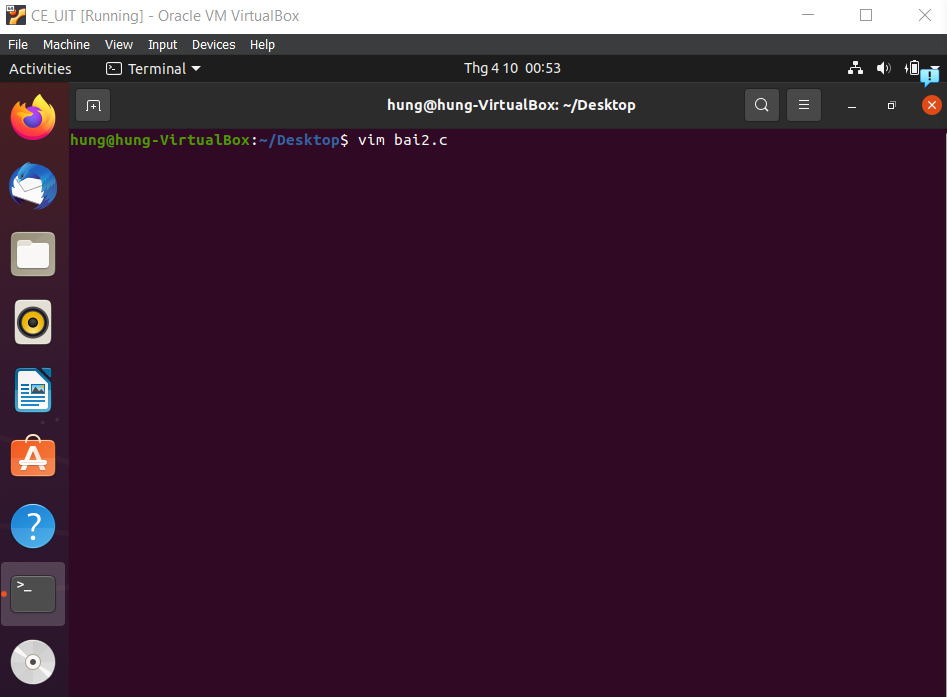
}

printf("I see %d coconuts!\n", num\_coconuts);

exit(0);

}

* Mở file bai2.c bằng vim editor. Nếu file bai2.c không tồn tại thì tạo file bai2.c.

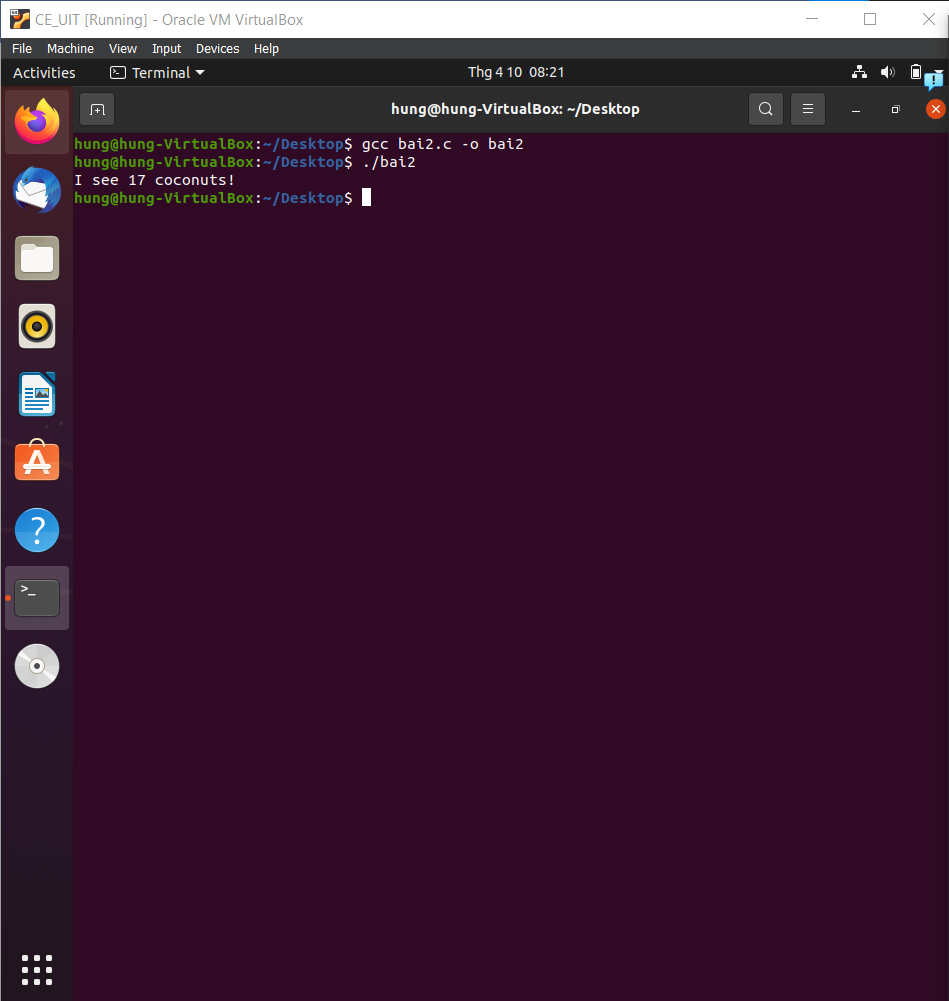


* Hình ảnh giao diện của vim editor khi gõ lại phần code mẫu vào.

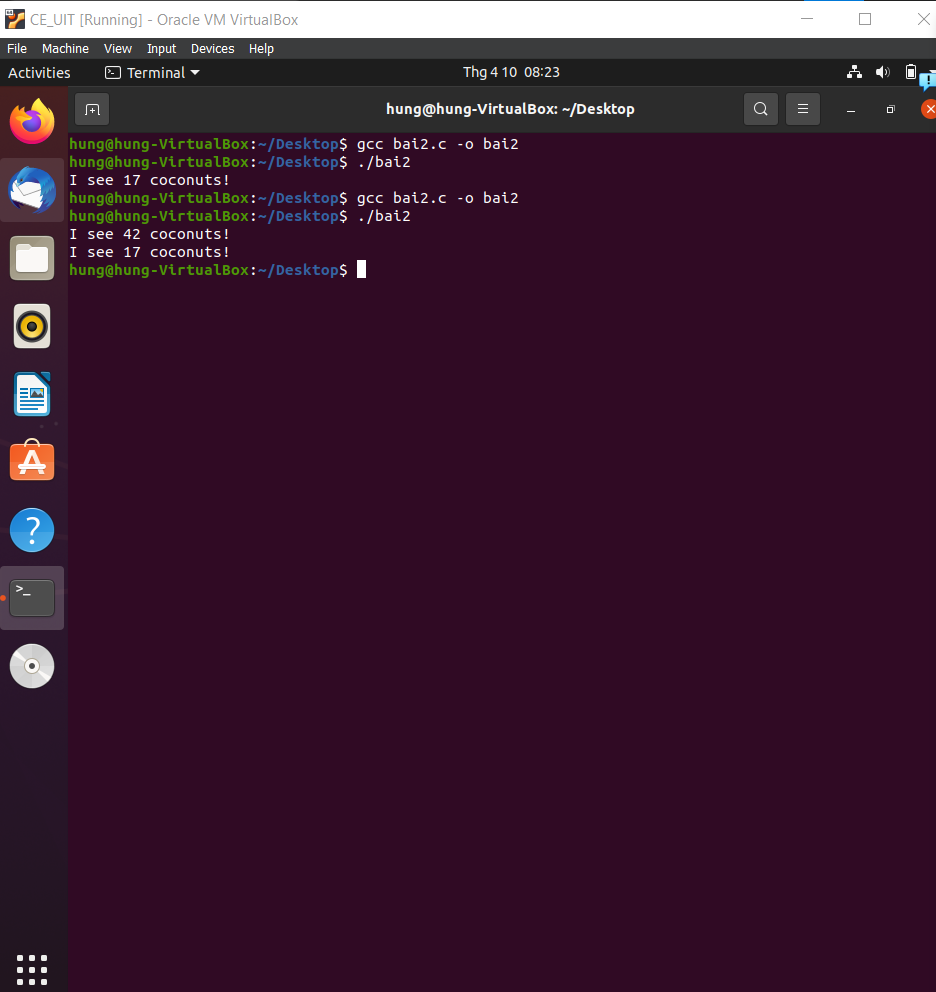
Text

Description automatically generated

* Kết quả của việc chạy chương trình :



* Khi tiến trình con chạy giá trị num\_coconut đã thay đổi thành giá trị 42, nhưng vì trong tiến trình có dòng lệnh exit(0); nên dòng lệnh in kết quả ra màn hình không được thực hiện, vì vậy nếu ta xóa dòng lệnh exit(0); sẽ có kết quả như sau:



**3. Trong phần thực hành, các ví dụ chỉ sử dụng thuộc tính mặc định của pthread, hãy tìm hiểu POSIX thread và trình bày tất cả các hàm được sử dụng để làm thay đổi thuộc tính của pthread, sau đó viết các chương trình minh họa tác động của các thuộc tính này và chú thích đầy đủ cách sử dụng hàm này trong chương trình. (Gợi ý các hàm liên quan đến thuộc tính của pthread đều bắt đầu bởi: pthread\_attr\_\*)**

- Các hàm được sử dụng để làm thay đổi thuộc tính của pthread:

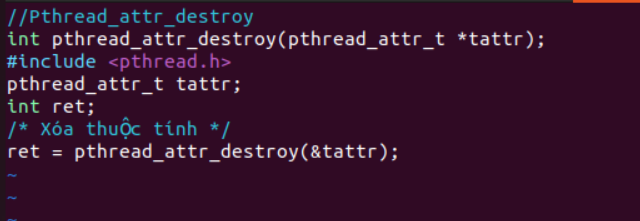
|  |  |
| --- | --- |
| **Hàm** | **Chức năng** |
| pthread\_attr\_init | Khởi tạo giá trị mặc định cho đối tượng thuộc tính |
| pthread\_attr\_destroy | Xóa bộ nhớ được cấp phát trong quá trình khởi tạo |
| pthread\_attr\_getschedparam | Trả về các tham số lịch trình (scheduling parameter) được xác định bởi pthread\_attr\_setschedparam() |
| pthread\_attr\_getschedpolicy | Để xuất scheduling policy của thread |
| pthread\_attr\_getdetachstate | Lấy truy xuất trạng thái khởi tạo của thread, có thể thể là riêng lẻ hoặc kết hợp |
| pthread\_attr\_getinheritsched | Trả về chính sách lịch trình (scheduling policy) được set bởi pthread\_attr\_setinheritsched (). |
| pthread\_attr\_getscope | Truy xuất phạm vi của thread |
| pthread\_attr\_setdetachstate | Sử dụng lại ID và tài nguyên của thread khi nó bị ngắt mà không phải chờ nếu thread có thuộc tính riêng lẻ |
| pthread\_attr\_setguardsize | Set kích thước của khu vực an toàn của thread |
| pthread\_attr\_setstackaddr | Set địa chỉ stack của thread |
| pthread\_attr\_setstacksize | Set kích thước stack của thread |
| pthread\_attr\_getguardsize | Lấy kích thước của khu vực an toàn của thread |
| pthread\_attr\_getstackaddr | Trả về địa chỉ stack của thread được set bởi pthread\_attr\_setstackaddr() |
| pthread\_attr\_getstacksize | Trả về kích thước stack của thread được set bởi pthread\_attr\_setstacksize(). |

**-** Các chương trình minh họa tác động của các thuộc tính này:

Ảnh có chứa văn bản

Mô tả được tạo tự động

*pthread\_attr\_init*



*pthread\_attr\_destroy*

Text

Description automatically generated

*pthread\_attr\_getschedparam*

Ảnh có chứa văn bản

Mô tả được tạo tự động

*pthread\_attr\_getschedpolicy*

Text

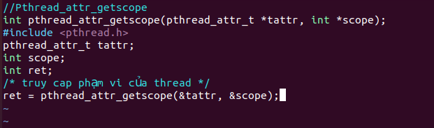
Description automatically generated

*pthread\_attr\_getdetachstate*

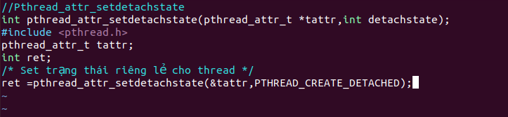
Ảnh có chứa văn bản

Mô tả được tạo tự động

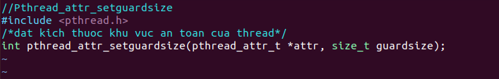
*pthread\_attr\_getinheritsched*



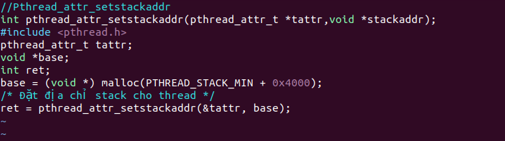
*pthread\_attr\_getscope*



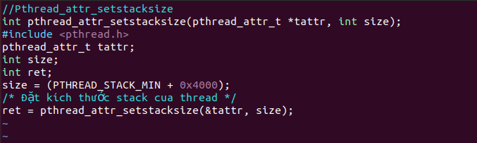
*pthread\_attr\_setdetachstate*



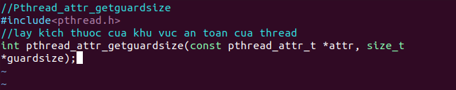
*pthread\_attr\_setguardsize*



*pthread\_attr\_setstackaddr*



*pthread\_attr\_setstacksize*

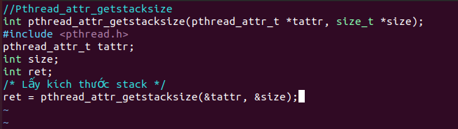


*pthread\_attr\_getguardsize*

Text

Description automatically generated

*pthread\_attr\_getstackaddr*

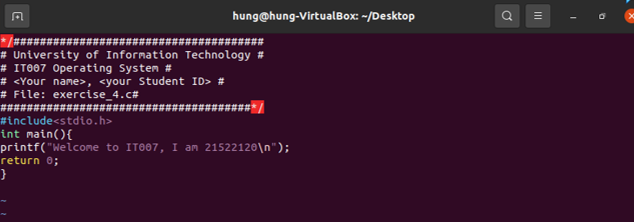


*pthread\_attr\_getstacksize*

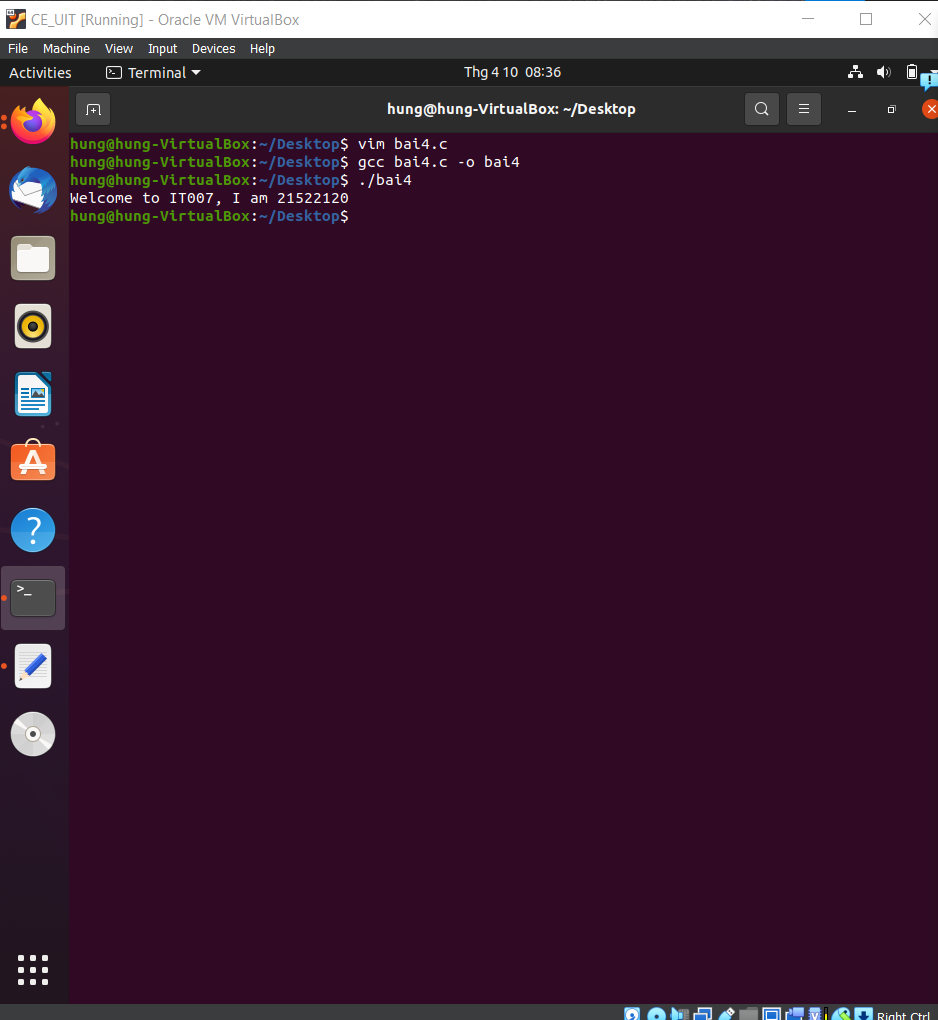
**4. Viết chương trình làm các công việc sau theo thứ tự:**

1. **In ra dòng chữ: “Welcome to IT007, I am <your\_Student\_ID>!”**

* Màn hình giao diện code để thực hiện bài toán:

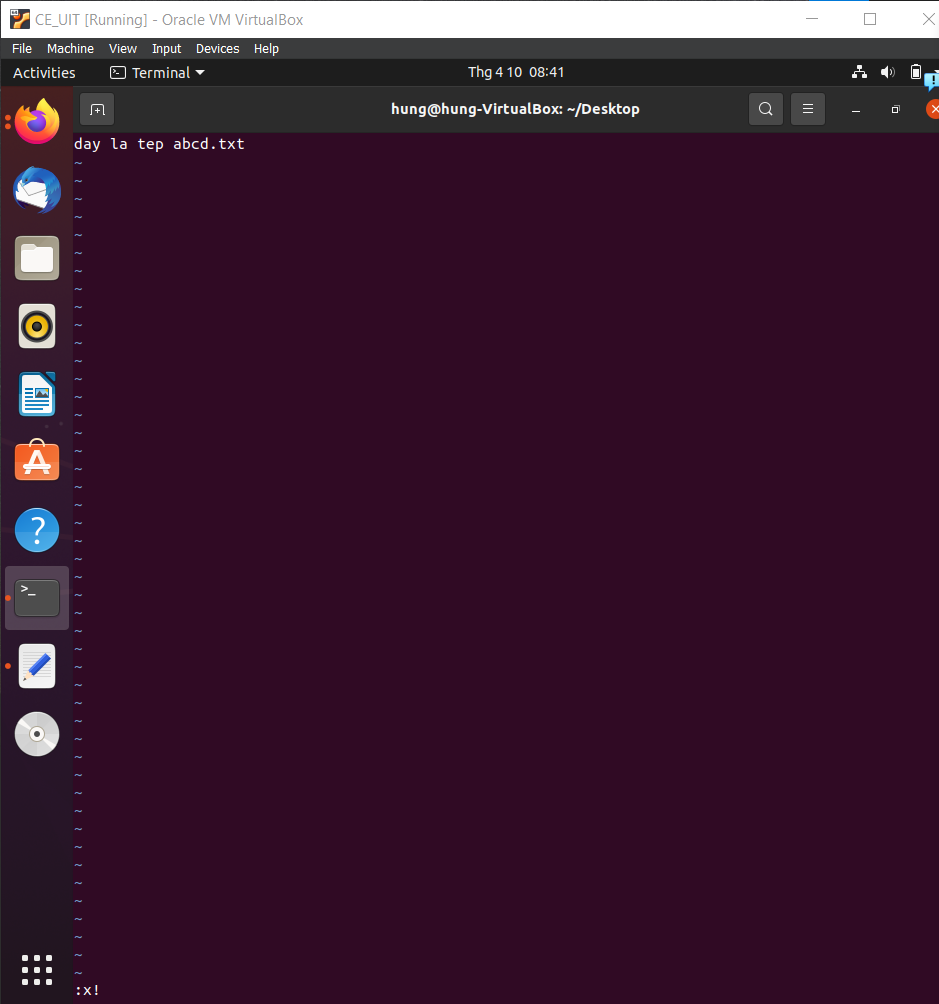


* Kết quả chạy chương trình của bài toán a



1. **Mở tệp abcd.txt bằng vim editor**

* Dùng lệnh vim abcd.txt để khởi tạo file abcd.txt. Soạn nội dung như hình, lưu và đóng file.



* Lúc này ta đã có file abcd.txt. Mở file bằng câu lệnh vim abcd.txt, ta được kết quả như hình.

A screenshot of a computer

Description automatically generated with medium confidence

1. **Tắt vim editor khi người dùng nhấn CTRL+C**Graphical user interface, text, application

   Description automatically generated
2. **Khi người dùng nhấn CTRL+C thì in ra dòng chữ: “You are pressed CTRL+C! Goodbye!”.**

* Chương trình để chạy bài:

Text

Description automatically generated

* Kết quả chạy chương trình:

Graphical user interface, text, application

Description automatically generated