|  |
| --- |
| Họ và tên:  Mã số sinh viên:  Lớp: |

HỆ ĐIỀU HÀNH  
BÁO CÁO LAB 3

**CHECKLIST**

**3.5. BÀI TẬP THỰC HÀNH**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **BT 1** | **BT 2** | **BT 3** | **BT 4** |
| **Trình bày cách làm** |  |  |  |  |
| **Chụp hình minh chứng** |  |  |  |  |
| **Giải thích kết quả** |  |  |  |  |

**3.6. BÀI TẬP ÔN TẬP**

|  |  |
| --- | --- |
|  | **BT 1** |
| **Trình bày cách làm** |  |
| **Chụp hình minh chứng** |  |
| **Giải thích kết quả** |  |

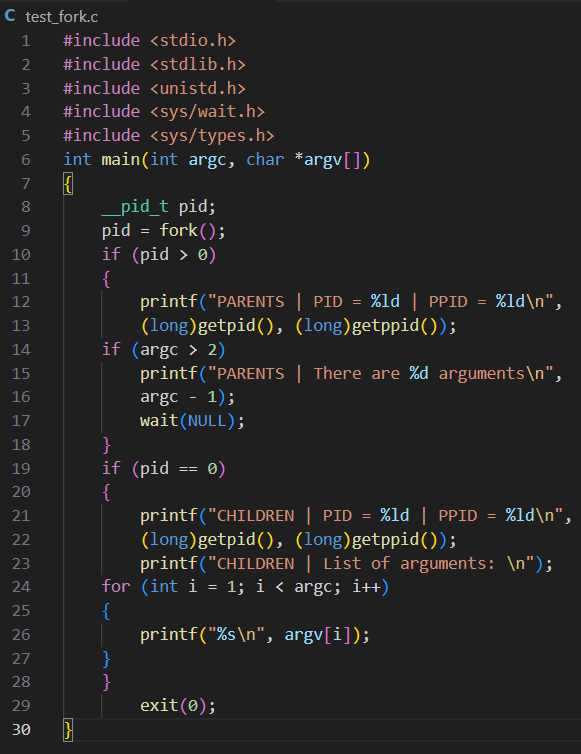
**Tự chấm điểm:**

*\*Lưu ý: Xuất báo cáo theo định dạng PDF, đặt tên theo cú pháp:* ***<MSSV>\_LAB3.pdf***

**2.5. BÀI TẬP THỰC HÀNH**

# Thực hiện Ví dụ 3-1, Ví dụ 3-2, Ví dụ 3-3, Ví dụ 3-4 giải thích code và kết quả nhận được?

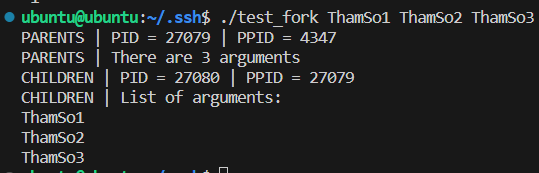
Ví dụ 3-1:



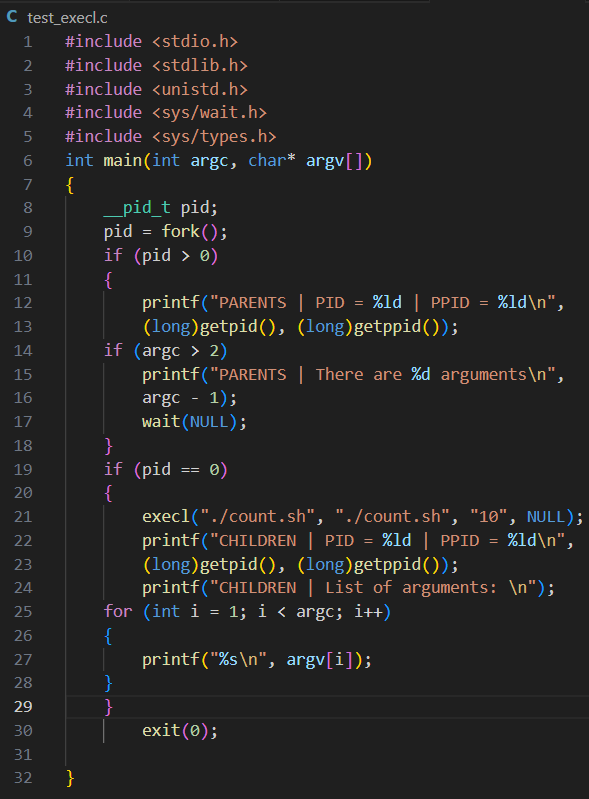
Giải thích code:

* 5 dòng include đầu tiên gồm tên các file và định nghĩa hàm, định nghĩa kiểu dữ liệu được sử dụng trong chương trình.
* Hàm main() là để đưa input vào các chương trình C. Trong hàm trên nhận 2 đối số: argc và argv. Đối số argc là đối số dòng lệnh được truyền cho chương trình. Đối số argv là một mảng các con trỏ đến các đối số dòng lệnh.
* Biến pid được sử dụng để lưu trữ PID của quy trình con.
* Hàm fork() tạo ra một quy trình con là một bản sao của quy trình cha, nhưng nó có PID và PPID riêng.
* Ở dòng lệnh if(pid > 0),

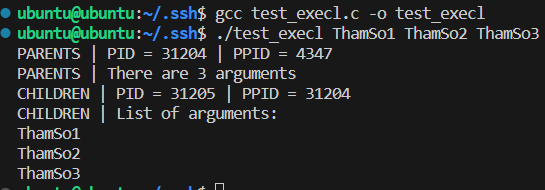
Kết quả sau khi thực thi:



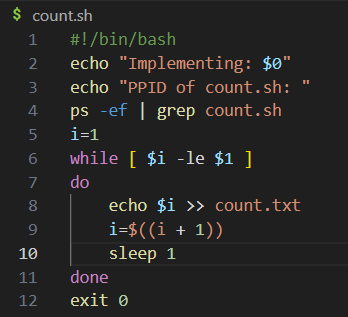
Ví dụ 3 – 2:

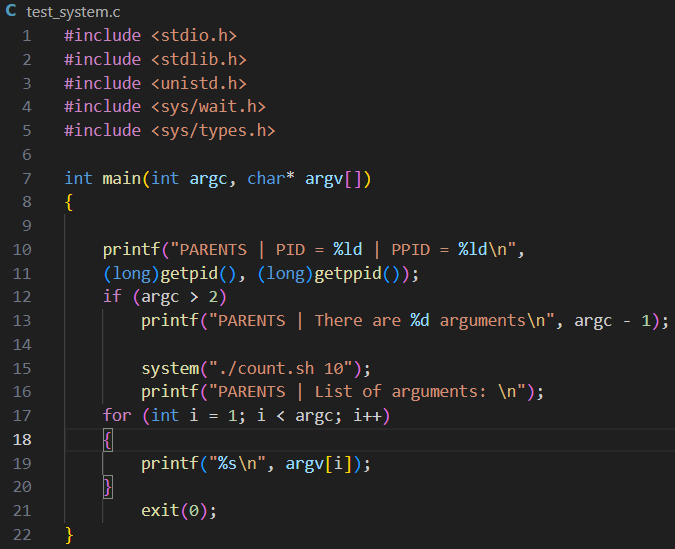


Kết quả thực thi:

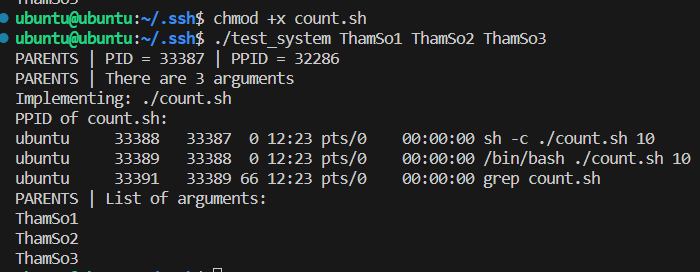


Ví dụ 3 – 3:



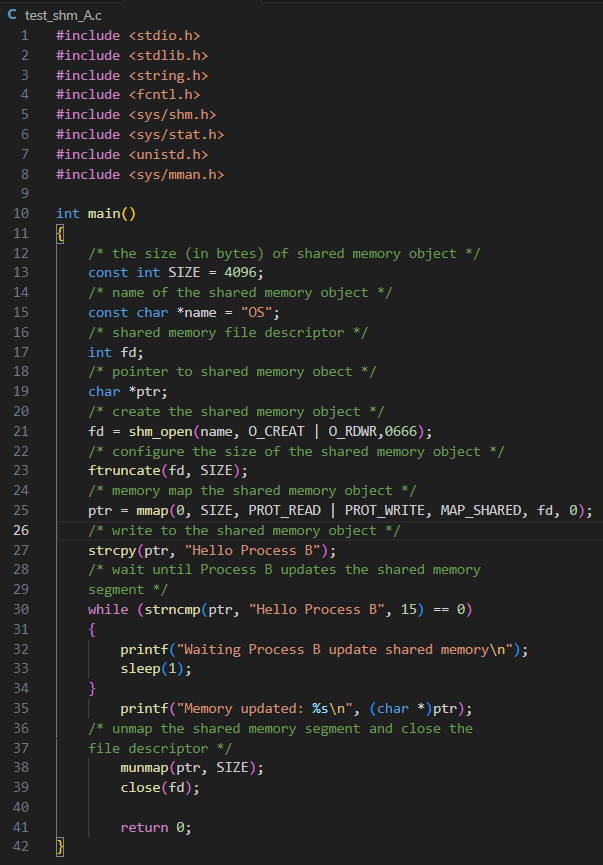


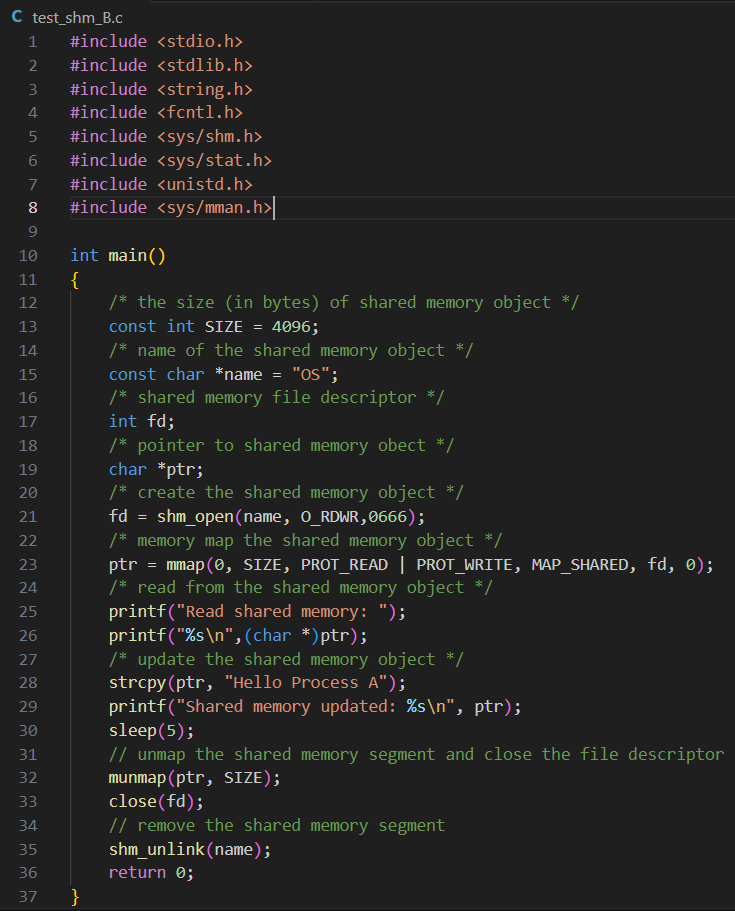
Kết quả thực thi:



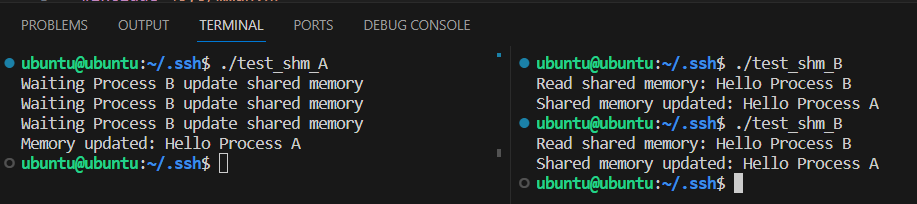
Ví dụ 3 – 4:

1. Process A:





Kết quả thực thi:



# Viết chương trình time.c thực hiện đo thời gian thực thi của một lệnh shell. Chương trình sẽ được chạy với cú pháp "./time <command>" với <command> là lệnh shell muốn đo thời gian thực thi.

Ví dụ:

$ ./time ls

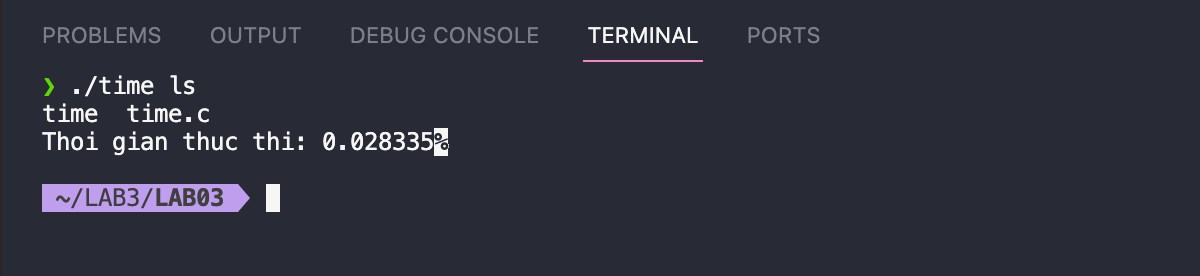
time.c

time

Thời gian thực thi: 0.25422

**Gợi ý: Tiến trình cha gọi hàm fork() tạo ra tiến trình con rồi wait(). Tiến trình con gọi hàm gettimeofday() để lấy mốc thời gian trước khi thực thi lệnh shell, sau đó sử dụng hàm execl() để thực thi lệnh. Sau khi tiến trình con kết thúc, tiến trình cha tiếp tục gọi hàm gettimeifday() một lần nữa để lấy mốc thời gian sau khi thực thi lệnh shell và tính toán.**





Giải thích:

* Khởi tạo các biến hằng để lưu trữ độ lớn của bộ nhớ chia sẻ và tên của bộ nhớ chia sẻ.
* Để chương trình có thể nhận các tham số khi thực thi, tiến hành truyền các tham số vào hàm main. argc là tham số thể hiện số tham số được truyền vào còn argv là mảng các tham số.
* Tạo tiến trình mới bằng hàm fork()
* Tạo biến time để lưu trữ giá trị thời gian.
* Khởi tạo biến fd để tạo vùng nhớ chia sẻ
* Khởi tạo biến con trỏ ptr để ánh xạ đến vùng nhớ chia sẻ
* Khởi tạo vùng nhớ được chia sẻ với lời gọi hệ thống shm\_open. Các tham số được truyền vào bao gồm tên vùng nhớ chia sẻ, các chế độ của vùng nhớ.
* Cài đặt độ lớn của vùng nhớ bằng hàm ftruncate.
* Tiến hành ánh xạ vùng nhớ tới biến ptr.
* Hàm if dùng để xác định tiến trình, nếu pid > 0 thì chương trình cha sẽ thực thi còn nếu pid = 0 thì tiến trình con sẽ thực thi.

**Tiến trình cha:**

▪ Tiến trình cha sẽ đợi đến khi nào tiến trình con thực thi xong tiến trình cha mới được thực thi bởi vì hàm wait(NULL)

▪ Lấy thời gian hiện tại bằng hàm gettimeofday(), thời gian này là thời gian sau khi tiến trình con đã thực thi xong.

▪ In ra màn hình kết quả bằng hàm printf, kết quả là thời gian sau thi tiến trình con thực thi xong trừ đi thời gian khi tiến trình con bắt đầu thực thi.

▪ Đóng các vùng nhớ chia sẻ

**Tiến trình con:**

▪ Lấy thời gian hiện tại bằng hàm gettimeofday(), thời gian này là thời gian lúc tiến trình con bắt đầu thực thi.

▪ Lưu thời gian vừa lấy được vào vùng nhớ chia sẻ để chia sẻ với tiến tình cha.

▪ Tạo biến cmd để lưu tên lệnh shell cần thực thi.

▪ Sử dụng hàm strcat để nối chuỗi cmd với lệnh shell cần thực thi được truyền vào thông qua hàm main.

▪ Sử dụng hàm execl để thực thi lệnh shell.

Giải thích kết quả thực thi:

* Kết quả thực thi với tham số được truyền vào là lệnh ls
* Chương trình đã thực thi lệnh ls và liệt kê ra những file có trong thư mục là file time và time.c
* Thời gian thực thi lệnh ls được đo và in ra màn hình

# Viết một chương trình làm bốn công việc sau theo thứ tự:

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

# Thực thi file script count.sh với số lần đếm là 120

# Trước khi count.sh đếm đến 120, bấm CTRL+C để dừng tiến trình này

# Khi người dùng nhấn CTRL+C thì in ra dòng chữ: “count.sh has stoppped”

Trả lời...

# Viết chương trình mô phỏng bài toán Producer - Consumer như sau:

# Sử dụng kỹ thuật shared-memory để tạo một bounded-buffer có độ lớn là 10 bytes.

# Tiến trình cha đóng vai trò là Producer, tạo một số ngẫu nhiên trong khoảng [10, 20] và ghi dữ liệu vào buffer

# Tiến trình con đóng vai trò là Consumer đọc dữ liệu từ buffer, in ra màn hình và tính tổng

# Khi tổng lớn hơn 100 thì cả 2 dừng lại

Trả lời...

**2.6. BÀI TẬP ÔN TẬP**

# Phỏng đoán Collatz xem xét chuyện gì sẽ xảy ra nếu ta lấy một số nguyên dương bất kỳ và áp dụng theo thuật toán sau đây: Phỏng đoán phát biểu rằng khi thuật toán này được áp dụng liên tục, tất cả số nguyên dương đều sẽ tiến đến 1. Ví dụ, với n = 35, ta sẽ có chuỗi kết quả như sau:

35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1  
Viết chương trình C sử dụng hàm fork() để tạo ra chuỗi này trong tiến trình con. Số bắt đầu sẽ được truyền từ dòng lệnh. Ví dụ lệnh thực thi ./collatz 8 sẽ chạy thuật toán trên n = 8 và chuỗi kết quả sẽ ra là 8, 4, 2, 1. Khi thực hiện, tiến trình cha và tiến trình con chia sẻ một buffer, sử dụng phương pháp bộ nhớ chia sẻ, hãy tính toán chuỗi trên tiến trình con, ghi kết quả vào buffer và dùng tiến trình cha để in kết quả ra màn hình. Lưu ý, hãy nhớ thực hiện các thao tác để kiểm tra input là số nguyên dương.

Trả lời...