Tên	Đinh Phương My
MSSV	52100703
Nhóm thực hành	N101

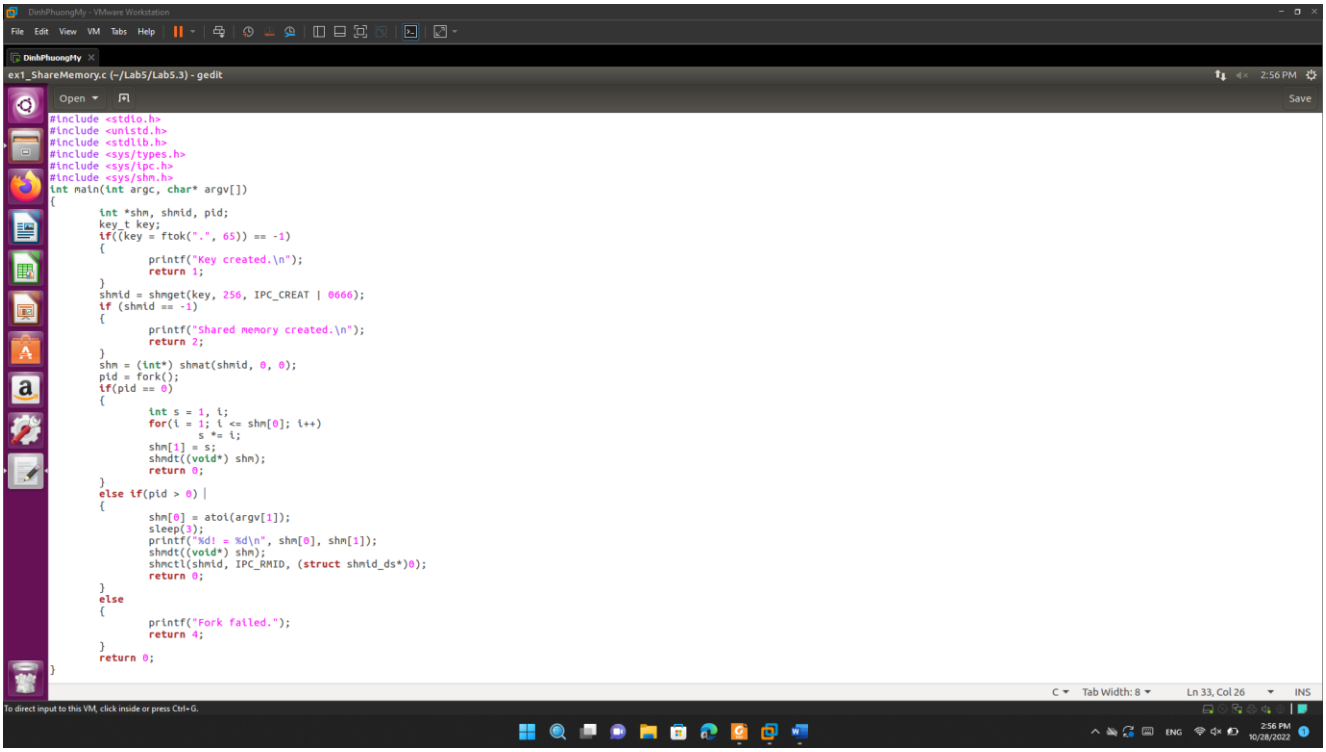
BÀI BÁO CÁO THỰC HÀNH LAB 5.3

Câu 1: Tiến trình cha chuyển đổi số đầu tiên (argv [1]) là một số nguyên lớn hơn 3 cho tiến trình con thông qua vùng nhớ chia sẻ. Tiến trình con nhận, tính giá trị n! = 1 * 2 * … * n và ghi nó vào vùng nhớ chia sẻ. Tiến trình cha nhận và xuất dữ liệu ra màn hình..

```
>./baitap2A.out 4
4! = 24
```

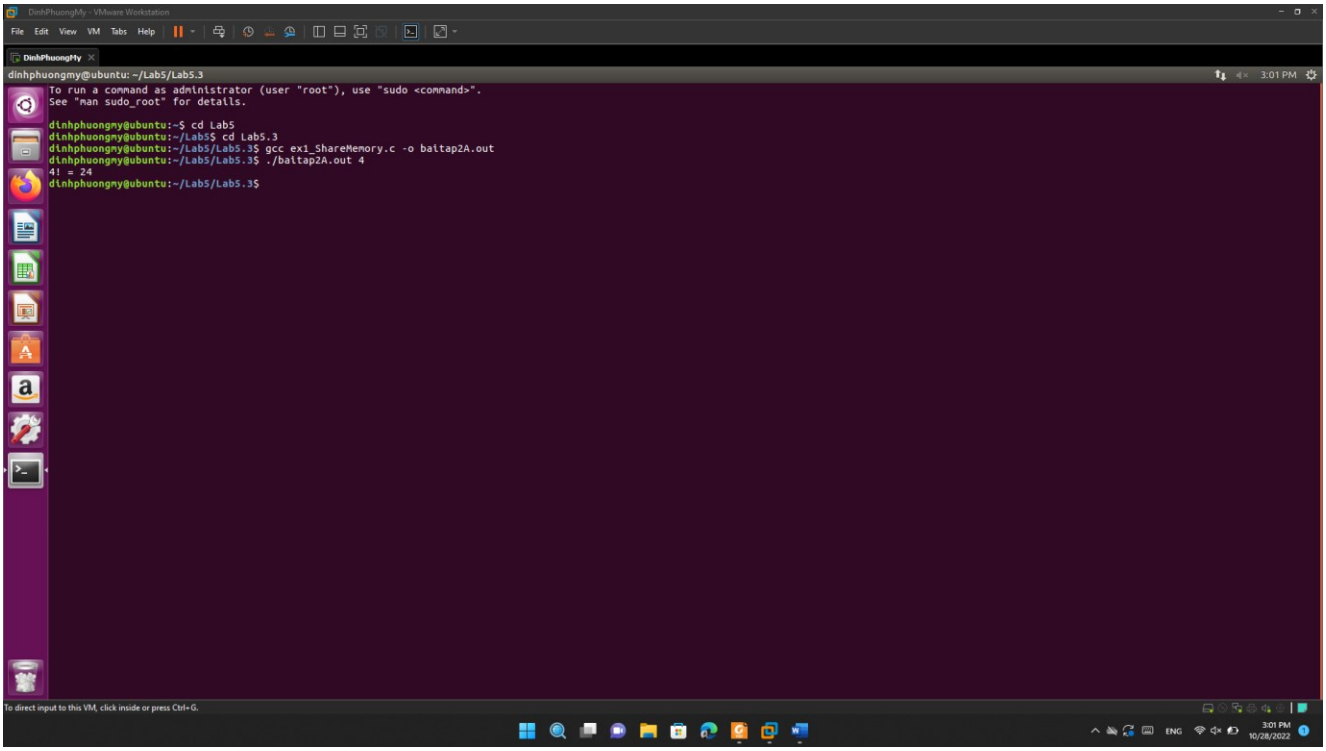
Share Memory

Code:



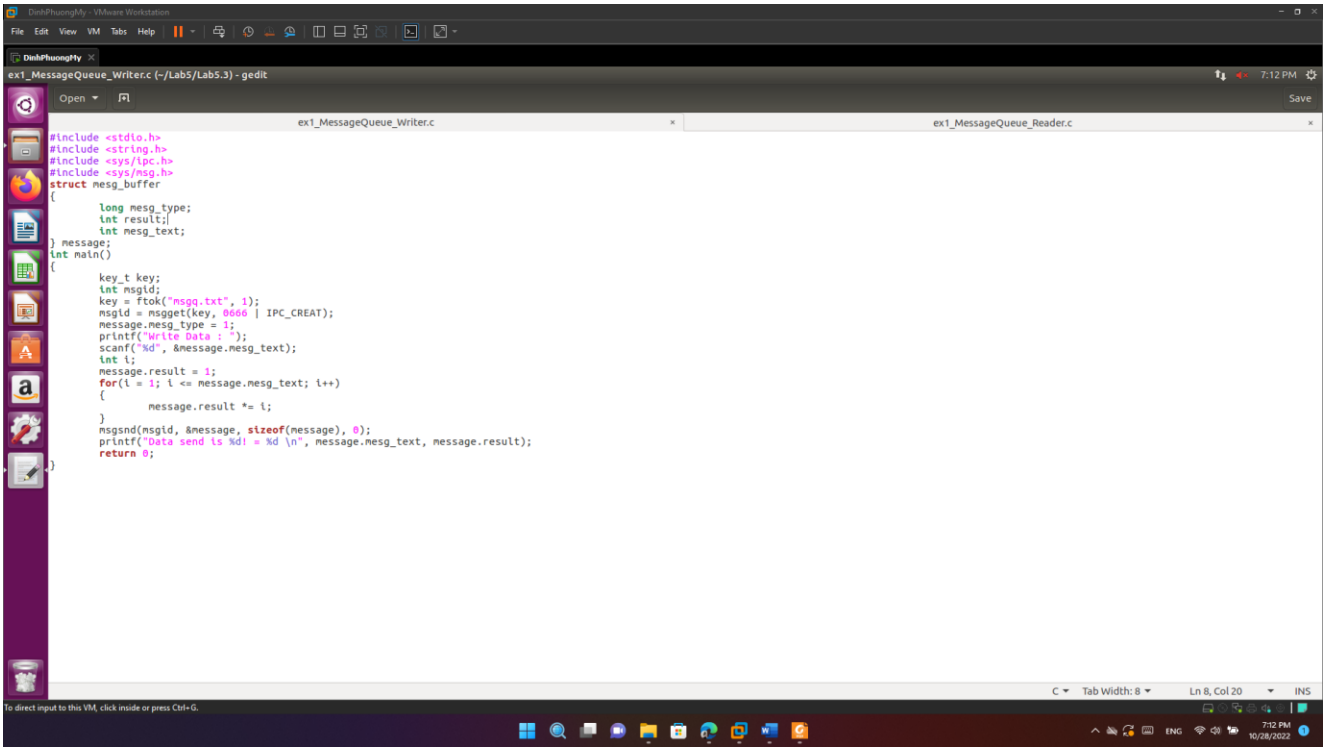
```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
int main(int argc, char* argv[])
{
        int *shm, shmid, pid;
        key_t key;
        if((key = ftok(".", 65)) == -1)
        {
                printf("Key created.\n");
                return 1;
        }
        shmid = shmget(key, 256, IPC_CREAT | 0666);
        if (shmid == -1)
        {
                printf("Shared memory created.\n");
                return 2;
        }
        shm = (int*) shmat(shmid, 0, 0);
        pid = fork();
        if(pid == 0)
        {
                int s = 1, i;
                for(i = 1; i <= shm[0]; i++)
                        s *= i;
                shm[1] = s;
                shmdt((void*) shm);
                return 0;
        }
        else if(pid > 0) |
        {
                shm[0] = atoi(argv[1]);
                sleep(3);
                printf("%d! = %d\n", shm[0], shm[1]);
                shmdt((void*) shm);
                shmctl(shmid, IPC_RMID, (struct shmid_ds*)0);
                return 0;
        }
        else
        {
                printf("Fork failed.");
                return 4;
        }
        return 0;
}
```

Run:

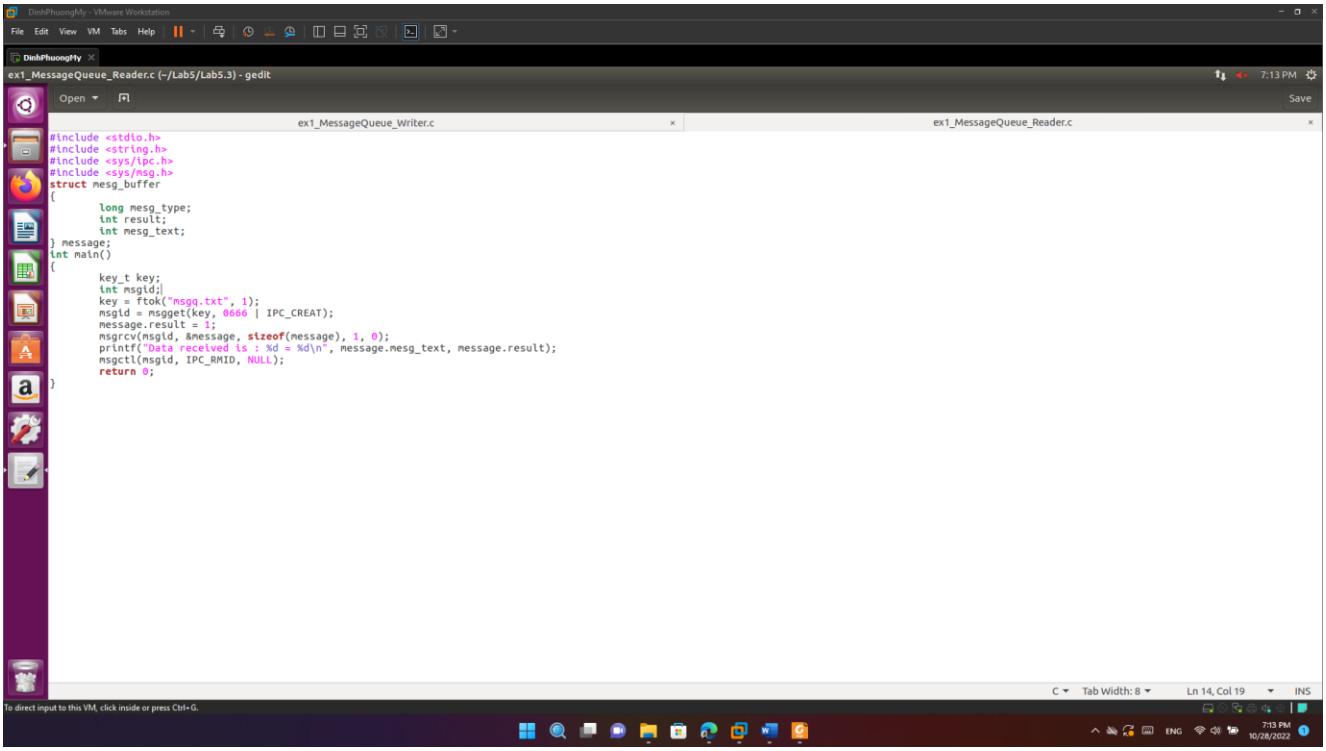


Message Queue

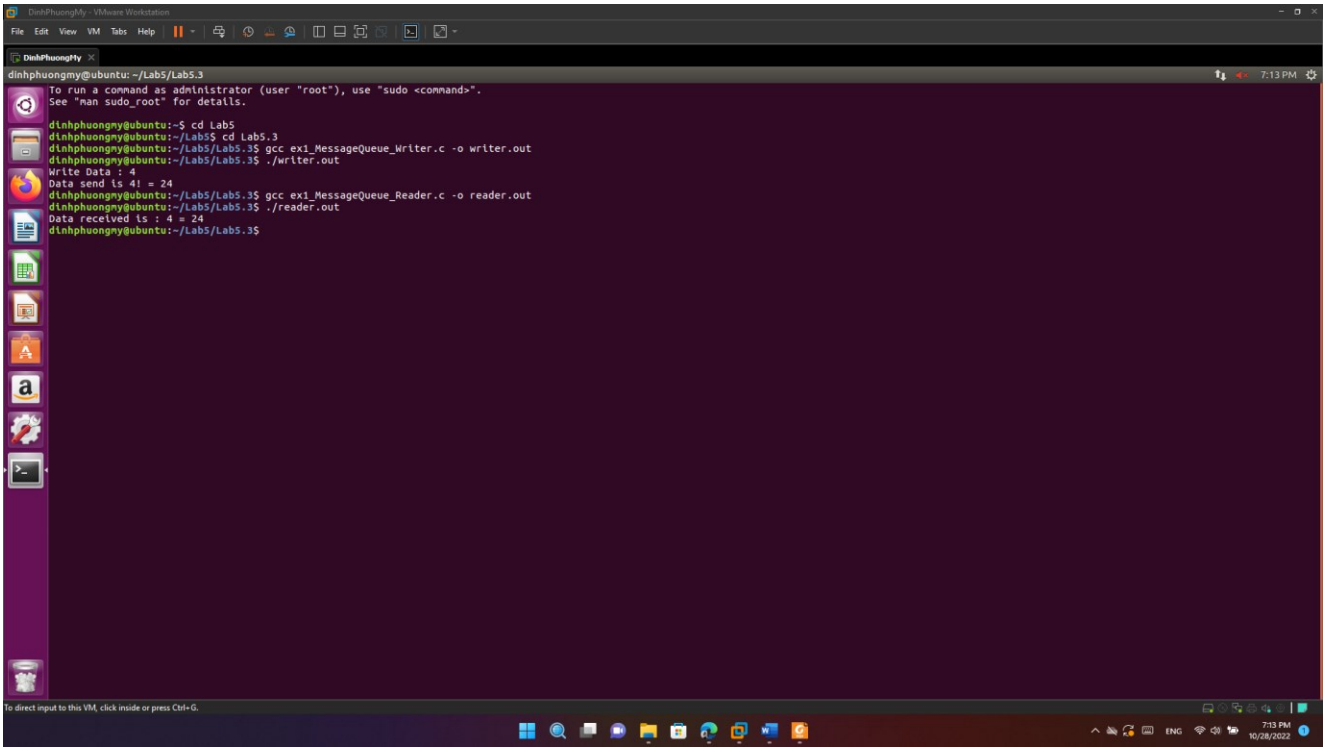
Writer



Reader:


ex1_MessageQueue_Reader.c (~/Lab5/Lab5.3) - gedit
Open
Save
ex1_MessageQueue_Writer.c
ex1_MessageQueue_Reader.c
#include <stdio.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/msg.h>
struct mesg_buffer
{
long mesg_type;
int result;
int mesg_text;
} message;
int main()
{
key_t key;
int msgid;
key = ftok("msgq.txt", 1);
msgid = msgget(key, 0666 | IPC_CREAT);
message.result = 1;
msgrcv(msgid, &message, sizeof(message), 1, 0);
printf("Data received is : %d = %d\n", message.mesg_text, message.result);
msgctl(msgid, IPC_RMID, NULL);
return 0;
}
C
Tab Width: 8
Ln 14, Col 19
INS

Run:

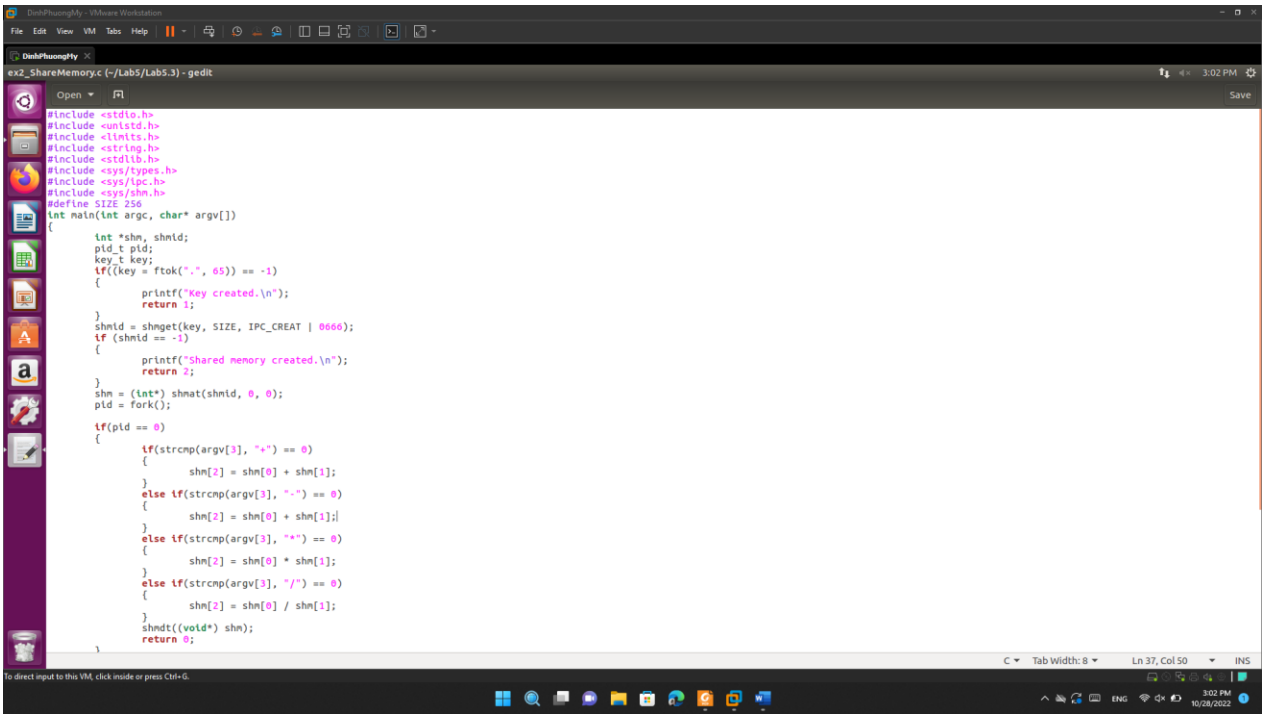

dinhphuongmy@ubuntu: ~/Lab5/Lab5.3
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
dinhphuongmy@ubuntu:~$ cd Lab5
dinhphuongmy@ubuntu:~/Lab5$ cd Lab5.3
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ gcc ex1_MessageQueue_Writer.c -o writer.out
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ ./writer.out
Write Data : 4
Data send is 4! = 24
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ gcc ex1_MessageQueue_Reader.c -o reader.out
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ ./reader.out
Data received is : 4 = 24
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$

Câu 2: Tiến trình cha đọc hai số nguyên và một thao tác +, -, *, / và chuyển tất cả cho tiến trình con. Tiến trình con tính toán kết quả và trả về cho tiến trình cha để in ra màn hình. Việc liên lạc sử dụng một vùng nhớ chia sẻ.

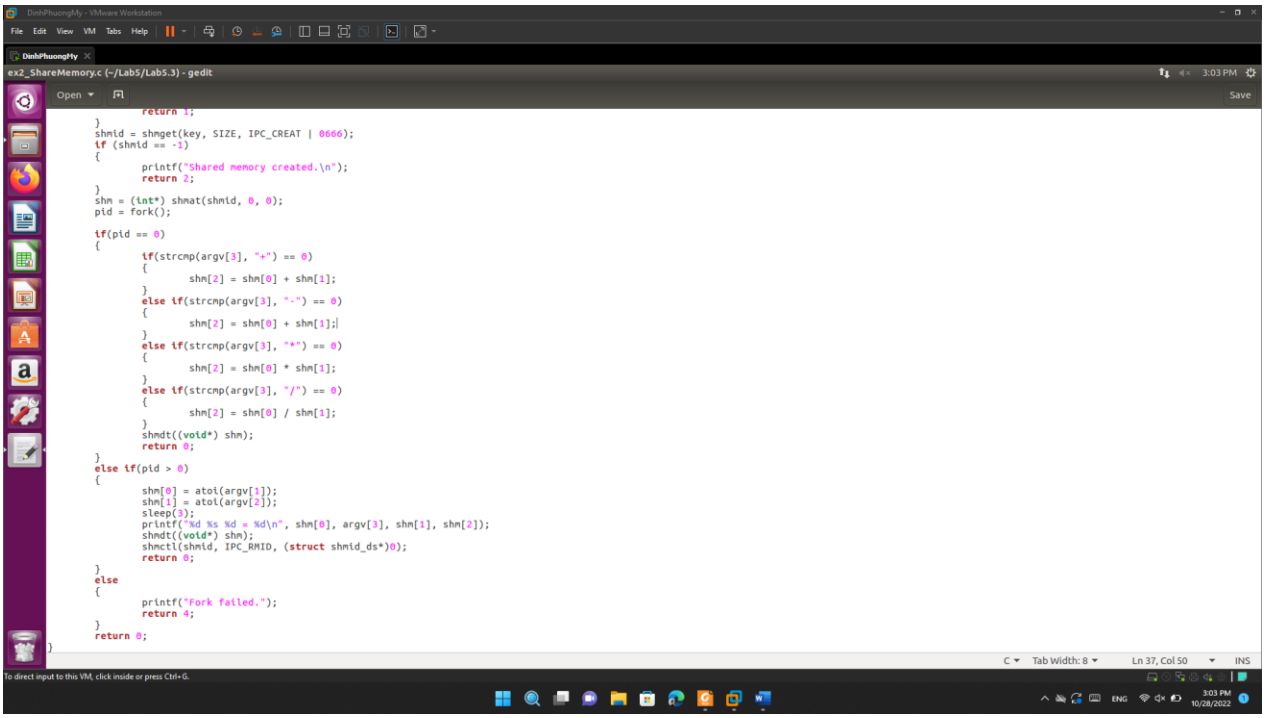
```
>./baitap3A.out 4 6 +
4 + 6 = 10
```

Share Memory

Code:

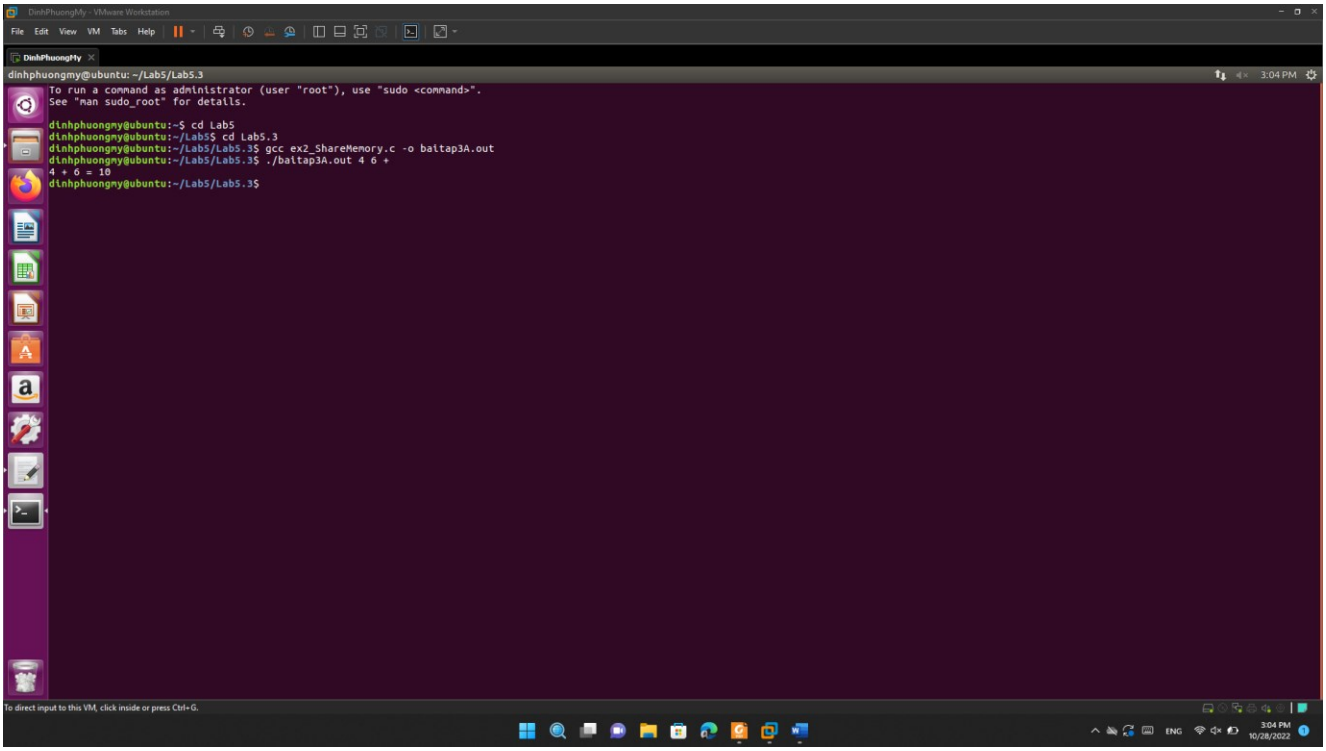


```
#include <stdio.h>
#include <unistd.h>
#include <limits.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#define SIZE 256
int main(int argc, char* argv[])
{
        int *shm, shmid;
        pid_t pid;
        key_t key;
        if((key = ftok(".", 65)) == -1)
        {
                printf("Key created.\n");
                return 1;
        }
        shmid = shmget(key, SIZE, IPC_CREAT | 0666);
        if (shmid == -1)
        {
                printf("Shared memory created.\n");
                return 2;
        }
        shm = (int*) shmat(shmid, 0, 0);
        pid = fork();

        if(pid == 0)
        {
                if(strcmp(argv[3], "+") == 0)
                {
                        shm[2] = shm[0] + shm[1];
                }
                else if(strcmp(argv[3], "-") == 0)
                {
                        shm[2] = shm[0] + shm[1];
                }
                else if(strcmp(argv[3], "*") == 0)
                {
                        shm[2] = shm[0] * shm[1];
                }
                else if(strcmp(argv[3], "/") == 0)
                {
                        shm[2] = shm[0] / shm[1];
                }
                shmdt((void*) shm);
                return 0;
        }
```



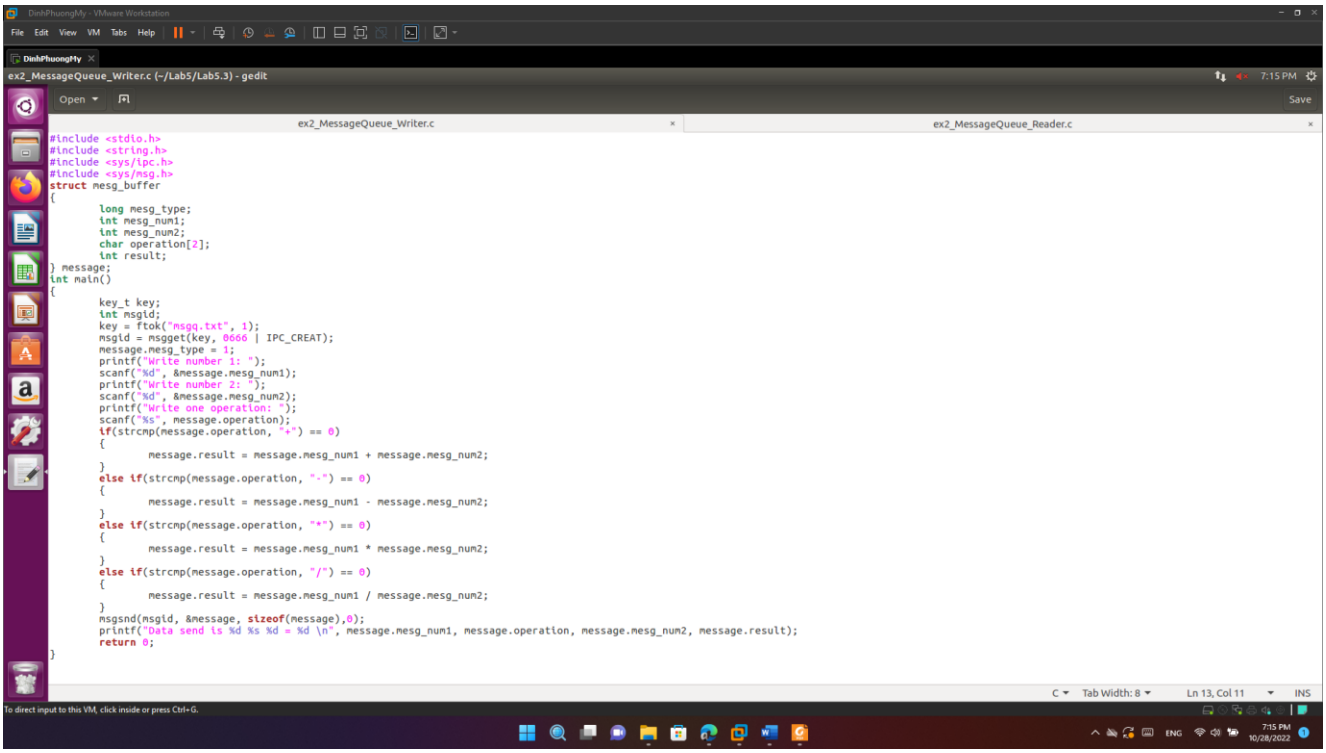
```
        return 1;
        }
        shmid = shmget(key, SIZE, IPC_CREAT | 0666);
        if (shmid == -1)
        {
                printf("Shared memory created.\n");
                return 2;
        }
        shm = (int*) shmat(shmid, 0, 0);
        pid = fork();

        if(pid == 0)
        {
                if(strcmp(argv[3], "+") == 0)
                {
                        shm[2] = shm[0] + shm[1];
                }
                else if(strcmp(argv[3], "-") == 0)
                {
                        shm[2] = shm[0] + shm[1];
                }
                else if(strcmp(argv[3], "*") == 0)
                {
                        shm[2] = shm[0] * shm[1];
                }
                else if(strcmp(argv[3], "/") == 0)
                {
                        shm[2] = shm[0] / shm[1];
                }
                shmdt((void*) shm);
                return 0;
        }
        else if(pid > 0)
        {
                shm[0] = atoi(argv[1]);
                shm[1] = atol(argv[2]);
                sleep(3);
                printf("%d %s %d = %d\n", shm[0], argv[3], shm[1], shm[2]);
                shmdt((void*) shm);
                shmctl(shmid, IPC_RMID, (struct shmid_ds*)0);
                return 0;
        }
        else
        {
                printf("Fork failed.");
                return 4;
        }
        return 0;
}
```

Run:

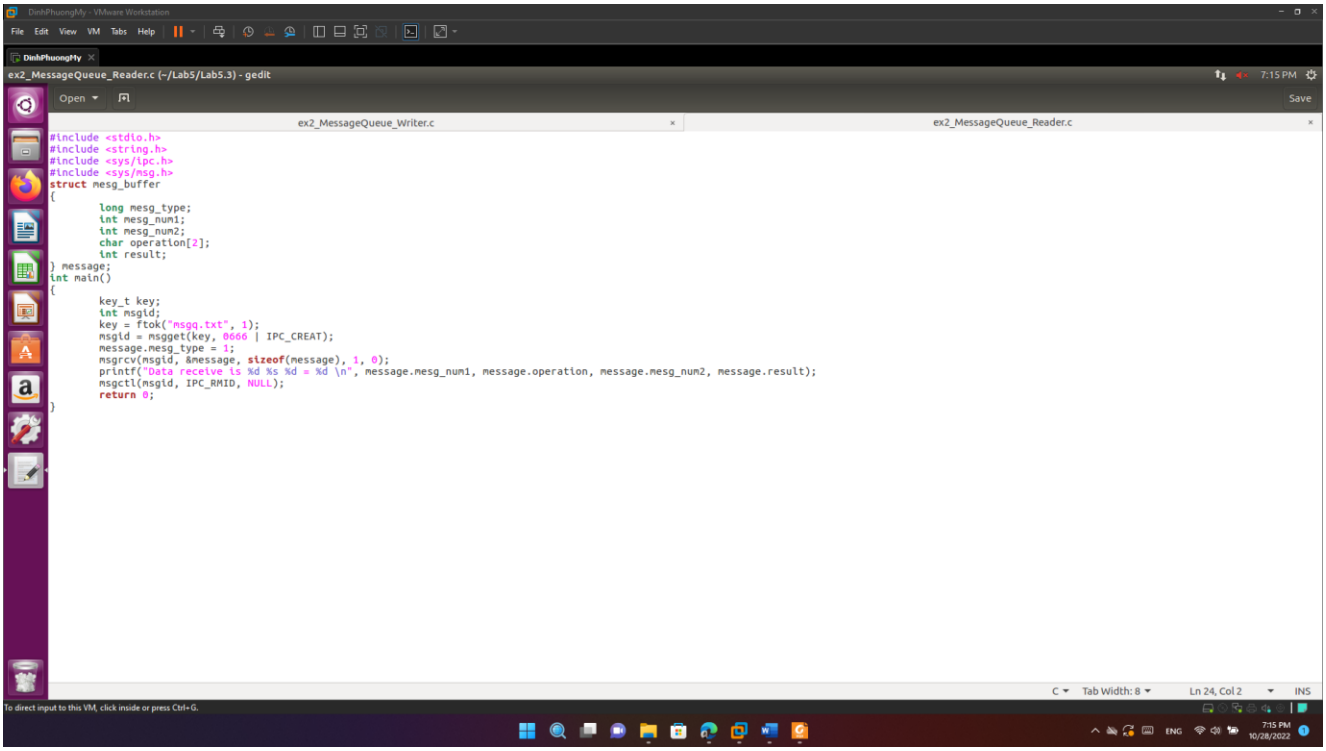


Message Queue

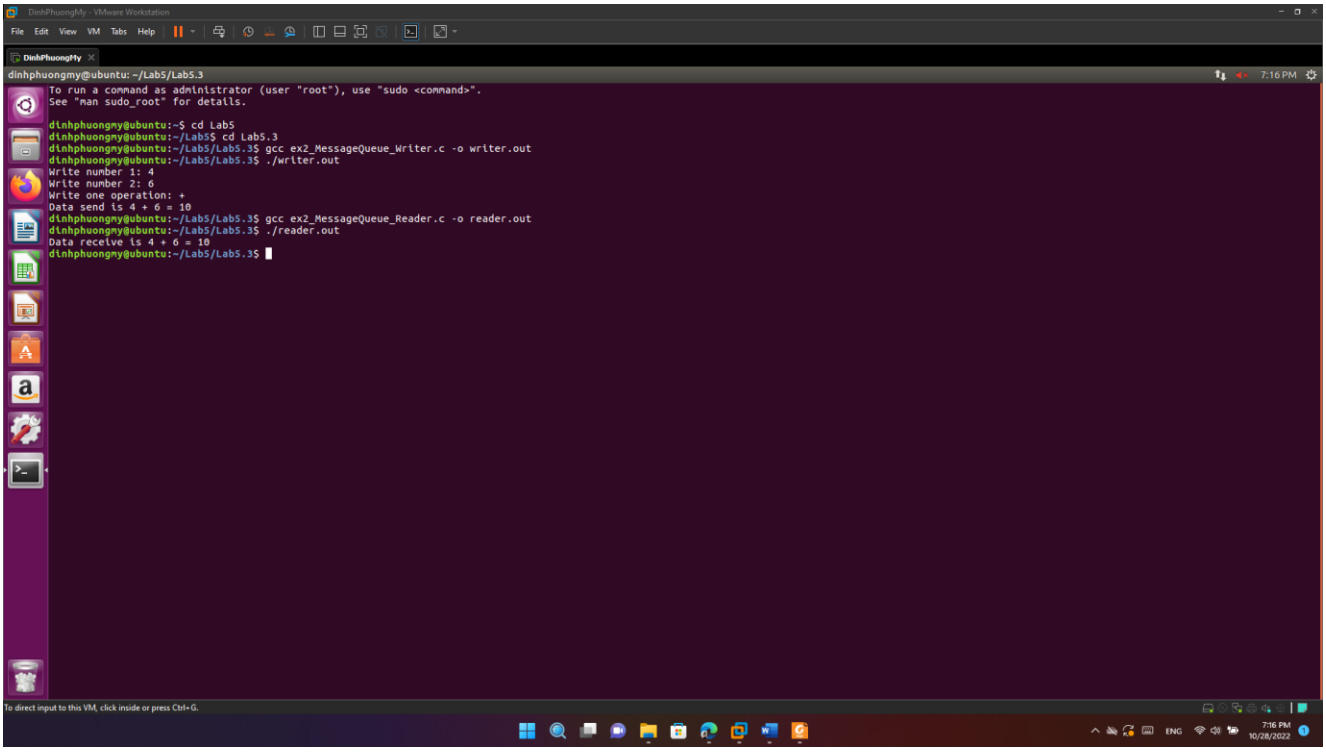
Writer



Reader:


ex2_MessageQueue_Reader.c (~/Lab5/Lab5.3) - gedit
Open
Save
ex2_MessageQueue_Writer.c
ex2_MessageQueue_Reader.c
#include <stdio.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/msg.h>
struct mesg_buffer
{
long mesg_type;
int mesg_num1;
int mesg_num2;
char operation[2];
int result;
} message;
int main()
{
key_t key;
int msgid;
key = ftok("msgq.txt", 1);
msgid = msgget(key, 0666 | IPC_CREAT);
message.mesg_type = 1;
msgrcv(msgid, &message, sizeof(message), 1, 0);
printf("Data receive is %d %s %d = %d \n", message.mesg_num1, message.operation, message.mesg_num2, message.result);
msgctl(msgid, IPC_RMID, NULL);
return 0;
}
C
Tab Width: 8
Ln 24, Col 2
INS

Run:


dinhphuongmy@ubuntu: ~/Lab5/Lab5.3
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
dinhphuongmy@ubuntu:~$ cd Lab5
dinhphuongmy@ubuntu:~/Lab5$ cd Lab5.3
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ gcc ex2_MessageQueue_Writer.c -o writer.out
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ ./writer.out
Write number 1: 4
Write number 2: 6
Write one operation: +
Data send is 4 + 6 = 10
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ gcc ex2_MessageQueue_Reader.c -o reader.out
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$ ./reader.out
Data receive is 4 + 6 = 10
dinhphuongmy@ubuntu:~/Lab5/Lab5.3$