

# LINUX INTERNALS IPC ASSIGNMENT

Vandit.Prajapati (150187)

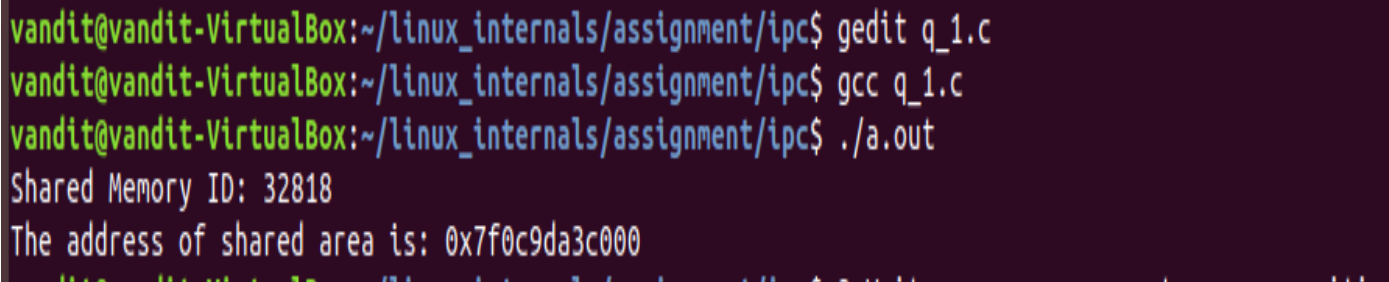
Q-1) Write a program that creates a shared memory region and displays shared memory id and also print address at which shared area in this process starts from.

```
#include<stdio.h>
#include<sys/ipc.h>
#include<sys/types.h>
#include<sys/shm.h>

int main()
{
    int id;
    char *addr;

    id=shmget(48, 250, IPC_CREAT|0644);

    printf("Shared Memory ID: %d\n",id);
    addr=shmat(id,0,0);
    printf("The address of shared area is: %p\n",addr);
    return 0;
}
```

A terminal window with a dark background and light-colored text. It shows the user 'vandit' at a 'vandit-VirtualBox' machine. The user is in the directory '~/linux\_internals/assignment/ipc'. They run 'gedit q\_1.c', then 'gcc q\_1.c', and finally './a.out'. The output of the program is 'Shared Memory ID: 32818' and 'The address of shared area is: 0x7f0c9da3c000'.

```
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ gedit q_1.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ gcc q_1.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ ./a.out
Shared Memory ID: 32818
The address of shared area is: 0x7f0c9da3c000
```

Q-2) Write a program that uses msgctl() and prints number of messages, number of bytes in message queue and also get Maximum number of bytes in queue for already existing message queue and also remove message queue at the end.

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/ipc.h>
#include<sys/msg.h>

struct msgbuf
{
    long mtype;
    char data[512];
};

int main(int argc, char *argv[])
{
    int qid;
    struct msqid_ds buf;

    struct msgbuf m;

    qid=msgget(32, IPC_CREAT|0644);
    printf("Message Queue ID: %d\n", qid);

    m.mtype=atoi(argv[1]);
    strcpy(m.data, argv[2]);
    msgsnd(qid, &m, strlen(m.data)+1, 0);

    msgctl(qid, IPC_STAT, NULL);
    printf("No. of messages: %hi\n", buf.msg_qnum);
    printf("No. of bytes in message queue: %hi\n", buf.msg_cbytes);
    printf("Max no. of bytes in message queue: %hi\n", buf.msg_qbytes);

    msgctl(qid, IPC_RMID, NULL);
}
```

```

vandit@vandit-VirtualBox:~/linux_internals/assignment/lpc$ gedit q_2.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/lpc$ gcc q_2.c
q_2.c: In function 'main':
q_2.c:25:2: warning: implicit declaration of function 'strcpy' [-Wimplicit-function-declaration]
  strcpy(m.data, argv[2]);
q_2.c:25:2: warning: incompatible implicit declaration of built-in function 'strcpy'
q_2.c:25:2: note: include '<string.h>' or provide a declaration of 'strcpy'
q_2.c:26:18: warning: implicit declaration of function 'strlen' [-Wimplicit-function-declaration]
  msgsnd(qid, &m, strlen(m.data)+1, 0);
q_2.c:26:18: warning: incompatible implicit declaration of built-in function 'strlen'
q_2.c:26:18: note: include '<string.h>' or provide a declaration of 'strlen'
q_2.c:29:29: warning: format '%hi' expects argument of type 'int', but argument 2 has type 'msgqnum_t {aka long unsigned int}' [-Wformat=]
  printf("No. of messages: %hi\n", buf.msg_qnum);
q_2.c:30:43: warning: format '%hi' expects argument of type 'int', but argument 2 has type '__syscall_ulong_t {aka long unsigned int}' [-Wformat=]
  printf("No. of bytes in message queue: %hi\n", buf.msg_cbytes);
q_2.c:31:47: warning: format '%hi' expects argument of type 'int', but argument 2 has type 'msglen_t {aka long unsigned int}' [-Wformat=]
  printf("Max no. of bytes in message queue: %hi\n", buf.msg_qbytes);
vandit@vandit-VirtualBox:~/linux_internals/assignment/lpc$ ./a.out 101 kernel
Message Queue ID: 2
No. of messages: 8560
No. of bytes in message queue: 8560
Max no. of bytes in message queue: -712

```

Q-3) Write a program parent process writing to pipe and child reading toggled data from pipe, and also analyse the data flow order from write end to read end.

```

#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>

```

```

int fds[2];

```

```

void parent_code()
{
    char *buf1="Vandit";
    char *buf2="Prajapati";
    close(fds[0]);
    write(fds[1],buf1,6);
    write(fds[1],buf2,9);
    printf("End of Parent\n");
    sleep(2);
}

```

```

void child_code()
{
    char buf[100];

```

```

    int n,i;
    close(fds[1]);
    n=read(fds[0],buf,100);
    printf("No of chars read = %d\n",n);
    for(i=0;i<n;i++)
    {
        printf("%c",buf[i]);
    }
    printf("\nEnd of child\n");
    sleep(2);
}

```

```

int main()
{
    int res,i;
    pid_t p;

    res=pipe(fds);

    if(res==-1)
    {
        perror("pipe");
        exit(1);
    }
    p=fork();
    if(p==0)
    {
        child_code();
    }
    else
    {
        parent_code();
    }
    return 0;
}

```

```
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ gedit q_2.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ gedit q_3.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ gcc q_3.c
vandit@vandit-VirtualBox:~/linux_internals/assignment/ipc$ ./a.out
End of Parent
No of chars read = 15
VanditPrajapati
End of child
```