

INFORME TAREA 1 – SISTEMAS OPERATIVOS

Integrantes: -Roberto Antonio Cruz Pinto

Fecha: 28-09-2025

Tarea 1: implementación de una Shell en C

Github: <https://github.com/Titoo-P/SO-Tarea1.git>

Informe:

El objetivo de esta tarea fue implementar una Shell en lenguaje C. El objetivo principal fue entender como funcionaba una Shell, aprendiendo a integrar y lograr de manera resumida:

- I. Mostrar un prompt
- II. Leer lo que el usuario escribe
- III. Ejecutar comandos con procesos hijos (fork, execvp, waitpid)
- IV. Manejar errores (caracteres, comandos que no existen, etc..)
- V. Implementar comando exit
- VI. Soportar pipes simples entre comandos

De esta manera aprender como el sistema operativo maneja procesos y la comunicación entre ellos.

Antes de continuar, debo aclarar que no logre encontrar otras personas que me añadieran a sus grupos (preguntando por sitios de WhatsApp y demás), voy a seguir buscando para la tarea 2, pero si no tengo suerte le escribo un correo para ver si se puede hacer algo.

Esta Tarea fue 100% hecho por mí, manteniendo ayudas de IA al mínimo (solo para errores de compilación que no lograba comprender) de esta manera poder aprender de mejor manera el curso, me confíe con la tarea y solamente puede completar la primera parte, la parte 2 no está hecha. A continuación, mostrare por ítems el desarrollo de la parte 1 de la tarea.

Item1 - Prompt:

Para mostrar el prompt, use `printf("mishell:$ ")` y `fflush(stdout)`. Con el

flush fue usado porque no aparecía en la consola hasta que se escribía algo, Así logre que se viera.

```
○ panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$
```

Item2 – Leer y parsear comandos:

Para leer y parsear comandos use getline (inclui biblioteca <sys/types.h>). getline para leer lo que escribe el usuario mas strtok para separar las líneas en “tokens” (comandos y sus argumentos). Se guardo el arreglo en un arreglo argv[] que después se pasa a “execvp”

```
● panto@LAPTOP-D5313MER:~/so/tarea1$ ls -l /
total 2748
lrwxrwxrwx   1 root root      7 Apr 22  2024 bin -> usr/bin
drwxr-xr-x   2 root root 4096 Feb 26  2024 bin.usr-is-merged
drwxr-xr-x   2 root root 4096 Apr 22  2024 boot
drwxr-xr-x  15 root root 3840 Sep 28 16:43 dev
drwxr-xr-x  90 root root 4096 Sep 28 16:44 etc
```

Item 3 – Comando interno exit:

Se agrego una condición simple que si se escribe exit, el programa hace break y se termina. Lo que es un comando interno de la Shell.

```
● panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$ ls
hola hola.c mishell mishell.c
mishell:$ exit
○ panto@LAPTOP-D5313MER:~/so/tarea1$
```


Item 4 – Manejo de error si el comando no existe:

Cuando un comando no existe, execvp devuelve el error. Coloque perror(“execvp”) para imprimir el error, de esta forma la shell no se cae, se muestra el error y sigue funcionando.

```
○ panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$ ksjdkksjdkksjd
execvp: No such file or directory
mishell:$
```

Item 5 – Enter vacio: Si el usuario apreta enter, verifico si la línea esta vacia y con “continue” vuelvo al bucle. De esta forma no se ejecuta

nada y solo se vuelve a mostrar el prompt.

```
panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$  Enter
mishell:$
```

Item 6 – Ejecucion con fork/exec/wait:

Con fork() se crea un hijo, en el hijo se llama a “execvp” con el comando y sus argumentos. En el padre uso “waitpid” para esperar a que el hijo termine. De esta forma se logra ejecutar comandos normales de Linux.

```
panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$ echo hola mundo
hola mundo
mishell:$
```

Item 7 – Pipes simples:

Implemente un pipe para que la salida de un comando se mande como entrada de otro. Use pipe(), dup2 y dos hijos con execvp. Probe con “hola | tr a-z A-Z” y funciona, mostrando el texto en mayusculas

```
panto@LAPTOP-D5313MER:~/so/tarea1$ ./mishell
mishell:$ echo hola mundo | tr a-z A-Z
HOLA MUNDO
mishell:$ ls -l | head -3
total 48
-rwxr-xr-x 1 panto panto 15960 Sep 26 18:17 hola
-rw-r--r-- 1 panto panto   98 Sep 26 18:17 hola.c
mishell:$
```

Pero no logre hacer que fuera de 3 comandos en pipes, solo logre que fueran de 2 comandos en pipes.