



UNS



DCIC

# SISTEMAS OPERATIVOS

Proyecto 2024



## Comisión 34

- Brenda Belen Martinez Ocampo
- Gonzalo Bernabe Di Marco

<b>1. Experimentación de Procesos y Threads con los Sistemas Operativos</b>	<b>3</b>
1.1. Procesos, threads y Comunicación	3
1. Pumper Nic.	3
2. Mini Shell.	3
1.2. Sincronización	5
1. Taller de Motos.	5
2. Santa Claus.	6
1. Inicialización	7
o sem_santa = 0 (Santa dormido)	7
<b>2. Problemas</b>	<b>8</b>
2.1. Lectura	8
2.2. Problemas Conceptuales	8
1. Paginación y Segmentación en Memoria	8
2. Tabla de páginas	11

# 1. Experimentación de Procesos y Threads con los Sistemas Operativos

## 1.1. Procesos, threads y Comunicación

### 1. Pumper Nic.

### 2. Mini Shell.

Esta Mini Shell es un programa de línea de comandos simple que permite al usuario ejecutar un conjunto limitado de comandos relacionados con la gestión de directorios y archivos en un sistema.

**- ayuda:** Muestra una lista de comandos disponibles y una breve descripción de cada uno cuando se llama a "help" proporcionando información sobre cómo utilizar la Mini Shell.

**-crearArchivo:** Crea un archivo en la ubicación actual utilizando el nombre que se le pasa como argumento.

**-crearDirectorio:** se copia el nombre del directorio de argv[1] a la variable dir, utilizando strncpy. Se declara un entero resultado y se intenta crear el directorio con la

función mkdir. El segundo parámetro (0777) establece los permisos para el nuevo directorio: rwxrwxrwx: permisos de lectura, escritura y ejecución para el propietario, el grupo y otros.

**-eliminarDirectorio:** se copia el nombre del directorio desde argv[1] a la variable dir, utilizando strncpy. se intenta eliminar el directorio utilizando la función rmdir. La función rmdir intenta eliminar el directorio especificado.

**-listarContenidoDirectorio:** Se declara un puntero dire de tipo struct dirent, que se utilizará para almacenar la información de cada entrada de directorio leída. se copia el nombre del directorio desde argv[1] a la variable dir, utilizando strncpy. Se declara un puntero d de tipo DIR y se intenta abrir el directorio especificado por dir utilizando la función opendir. Si el directorio se abre correctamente, se entra en un bucle que utiliza readdir para leer cada entrada del directorio.

readdir devuelve un puntero a la estructura dirent que contiene información sobre la entrada, o NULL si no hay más entradas.

Una vez que se han leído todas las entradas, se cierra el directorio con closedir(d)

**-modificarPermisos:** Se utiliza stat para obtener la información sobre el archivo o directorio cuyo nombre se proporciona en argv[1]. Se inicializa nuevoPermiso con los permisos actuales del archivo. Finalmente, se aplica el nuevo conjunto de permisos utilizando chmod

**-mostrarContenidoArchivo:** se copia el nombre del archivo desde argv[1] a la variable dir, utilizando strncpy. se intenta abrir el archivo especificado por dir en modo lectura ("r"). se utiliza un bucle while junto con fgets para leer el contenido del archivo línea por línea. Una vez que se han leído todas las líneas o si el archivo no se abre, se cierra el archivo con fclose(f).

**-minishell:** char comando[MAXIMO]: Almacena el comando ingresado por el usuario.

char d[MAXIMO]: Almacena argumentos adicionales para los comandos (como nombres de directorios o archivos).

char val[100]: Almacena parámetros de permisos para el comando chmod

Un bucle while que continuará ejecutándose hasta que el usuario ingrese "exit". Se solicita al usuario que ingrese un comando y se almacena en la variable comando. Se comparan los comandos ingresados usando strcmp y se ejecutan acciones correspondientes para cada uno. Para cada comando, se utiliza fork() para crear un nuevo proceso hijo. En el proceso hijo, se llama a execv() para ejecutar el programa correspondiente. El proceso padre espera a que el proceso hijo termine su ejecución.

### Capturas de casos prueba de la minishell:

- 1) Primero nos posicionamos en donde esta ubicada la minishell.
- 2) Cambiamos el permiso a ejecutable del script generado.
- 3) Ejecutamos el script
- 4) La minishell nos muestra el mensaje "Bienvenido a la Mini Shell: Para ver los comandos utilice 'help'".

- 5) ingresamos help.
- 6) la minishell nos muestra el mensaje de que comandos podemos usar y una breve descripción de para que sirve cada uno.

```
brendam@raspberrypi: ~/Desktop/Minishell
File Edit Tabs Help
brendam@raspberrypi:~ $ cd /home/brendam/Desktop/Minishell
brendam@raspberrypi:~/Desktop/Minishell $ chmod +x ejecutar.sh
brendam@raspberrypi:~/Desktop/Minishell $ ./ejecutar.sh
Bienvenido a la Minishell: Para ver los comandos utilice 'help'.

-> help
-mkdir nombre: Crea un directorio con el nombre indicado en la ubicacion actual.
-rmdir nombre : Elimina el directorio con ese nombre indicado.
-touch nombre: Crea un archivo con el nombre indicado en la dirección actual.
-o puede indicar el directorio donde crear el archivo de la forma [DIRECTORIO]/[NOMBRE].
-ls nombre: Lista el contenido del directorio con el nombre indicado.
-cat nombre: Muestra el contenido del archivo con el nombre indicado.
-chmod nombre permisos : Cambia los permisos indicados de un archivo en el directorio con ese nombre.
Los comandos de los permisos son:
    Para escritura: '+escritura' 0 '-escritura'
    Para lectura: '+lectura' 0 '-lectura'
    Para ejecucion: '+ejecutar' 0 '-ejecutar'
-exit : Sale de la Minishell.

-> █
```

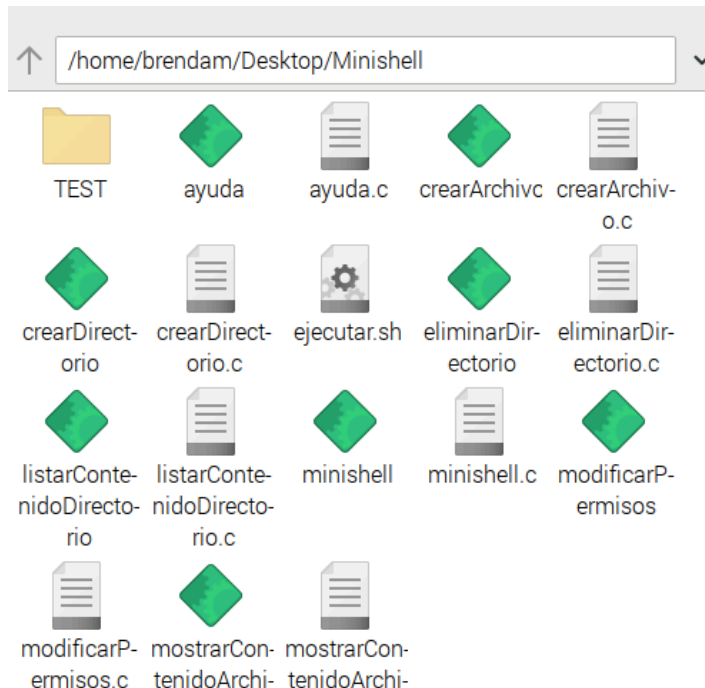
- 7) ingresamos el comando mkdir TEST y nos muestra el mensaje de que se creo correctamente el directorio test (se puede ver en la captura siguiente al comando) Al querer crear el mismo directorio nos muestra el mensaje de error ya que el directorio ya existe.

```
brendam@raspberrypi: ~/Desktop/Minishell
File Edit Tabs Help
brendam@raspberrypi:~/Desktop/Minishell $ ./ejecutar.sh
Bienvenido a la Minishell: Para ver los comandos utilice 'help'.

-> help
-mkdir nombre: Crea un directorio con el nombre indicado en la ubicacion actual.
-rmdir nombre : Elimina el directorio con ese nombre indicado.
-touch nombre: Crea un archivo con el nombre indicado en la dirección actual.
-o puede indicar el directorio donde crear el archivo de la forma [DIRECTORIO]/[NOMBRE].
-ls nombre: Lista el contenido del directorio con el nombre indicado.
-cat nombre: Muestra el contenido del archivo con el nombre indicado.
-chmod nombre permisos : Cambia los permisos indicados de un archivo en el directorio con ese nombre.
Los comandos de los permisos son:
    Para escritura: '+escritura' 0 '-escritura'
    Para lectura: '+lectura' 0 '-lectura'
    Para ejecucion: '+ejecutar' 0 '-ejecutar'
-exit : Sale de la Minishell.

-> mkdir TEST
Directorio creado correctamente
-> mkdir TEST
Error al crear el directorio: File exists

-> █
```



8) Ingresamos rmdir TEST y vemos el mensaje de que se ha eliminado correctamente y efectivamente se elimina.

Al querer hacer nuevamente rmdir TEST nos muestra el error de que el directorio no existe.

```

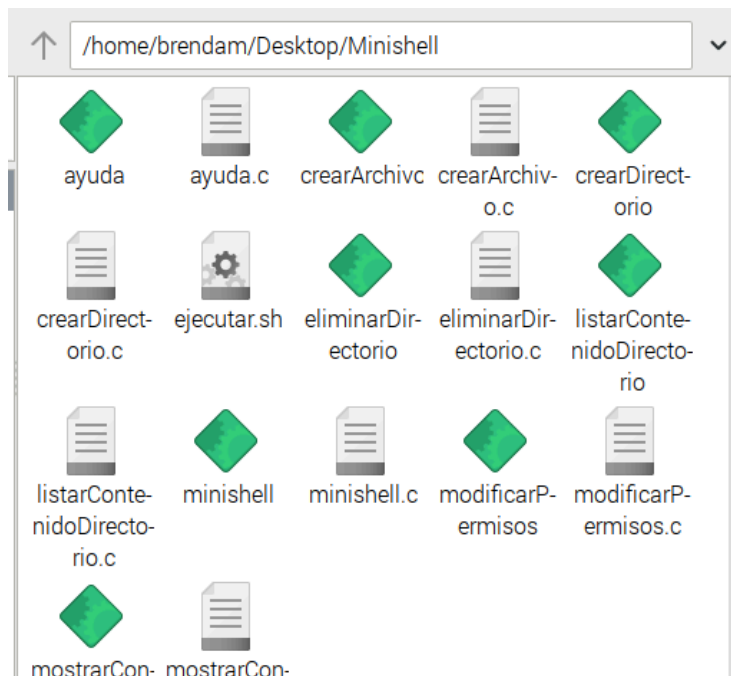
brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help
-o puede indicar el directorio donde crear el archivo de la forma [DIRECTORIO]/[NOMBRE].
-ls nombre: Lista el contenido del directorio con el nombre indicado.
-cat nombre: Muestra el contenido del archivo con el nombre indicado.
-chmod nombre permisos : Cambia los permisos indicados de un archivo en el directorio con ese nombre.
Los comandos de los permisos son:
    Para escritura: '+escritura' O '-escritura'
    Para lectura: '+lectura' O '-lectura'
    Para ejecucion: '+ejecutar' O '-ejecutar'
-exit : Sale de la Minishell.

-> mkdir TEST
Directorio creado correctamente
-> mkdir TEST
Error al crear el directorio: File exists

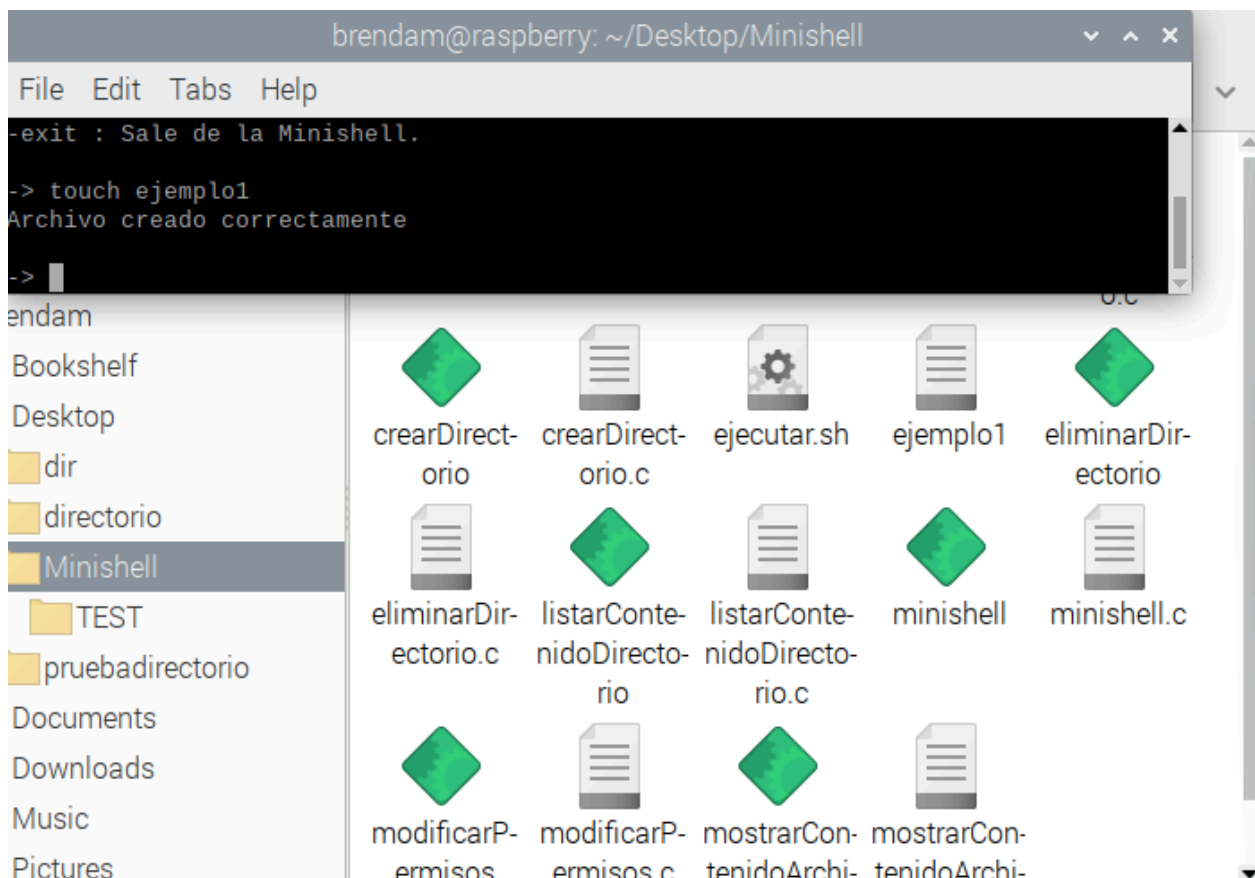
-> rmdir TEST
Directorio eliminado correctamente
-> rmdir TEST
Error al eliminar el directorio

: No such file or directory
->

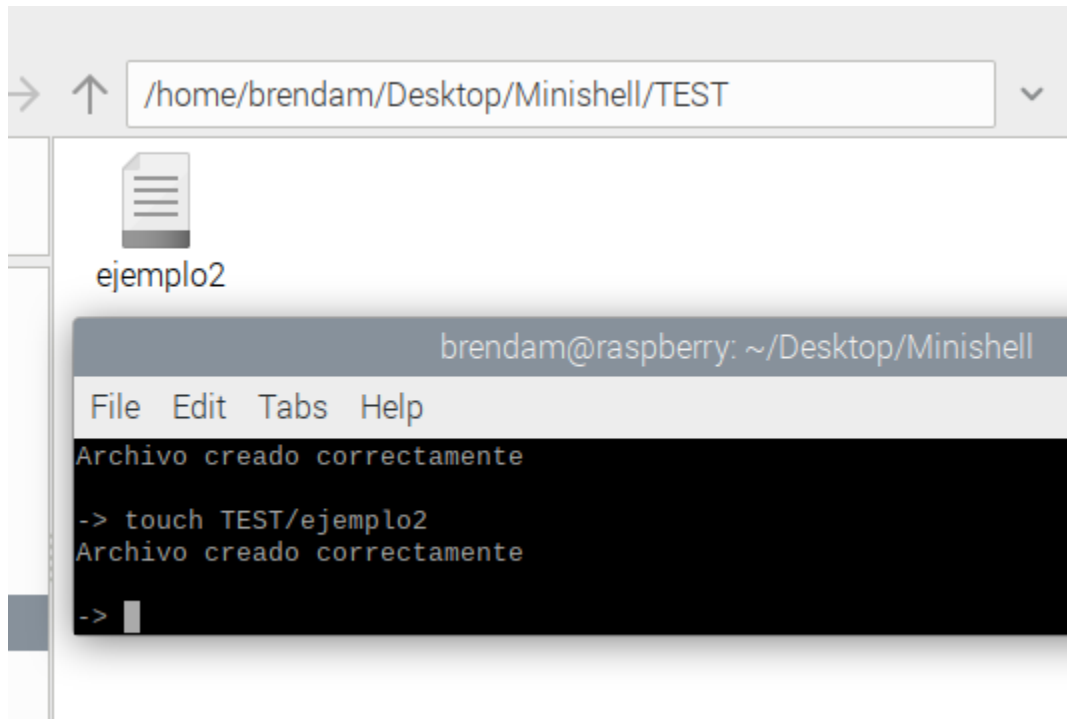
```



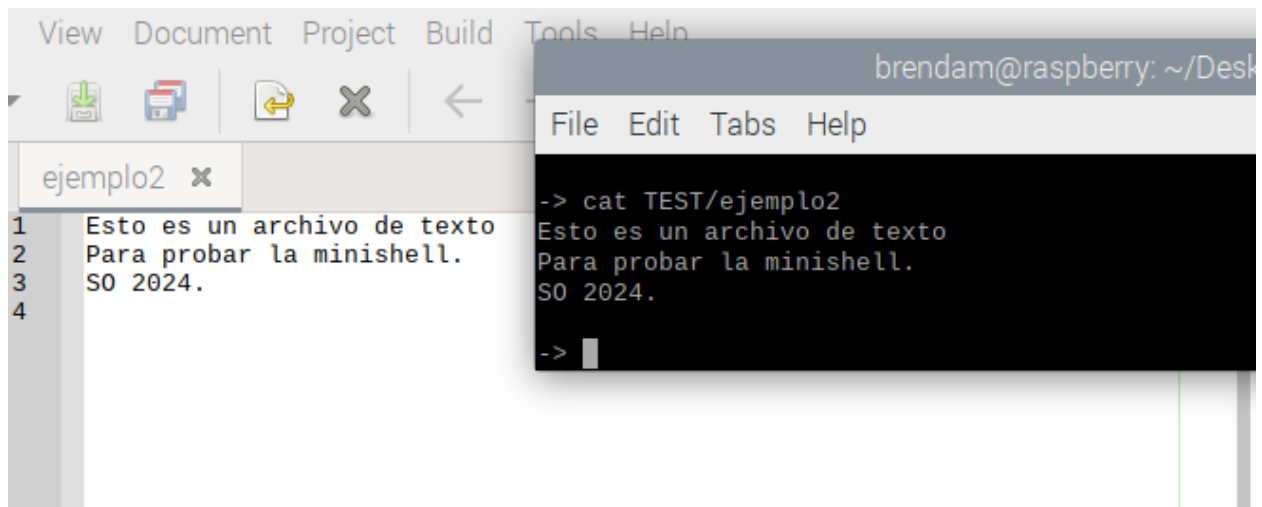
- 9) al ingresar el comando touch ejemplo1 vemos como se crea un archivo en la direccion actual.



Podemos hacer touch TEST/ejemplo2 para crear un archivo en una direccion especificada, en este caso TEST, y vemos como se crea correctamente.

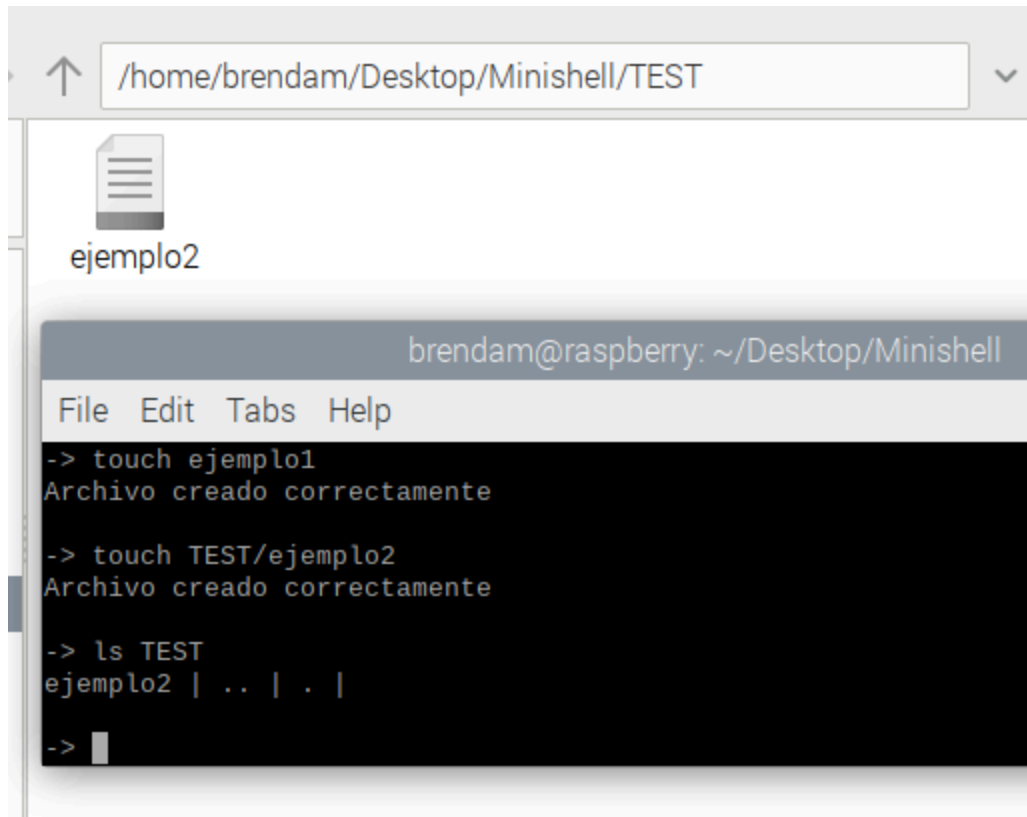


10) Al ingresar `cat TEST/ejemplo2` vemos que nos muestra el contenido del archivo `ejemplo2`.



11) Ingresamos el comando `ls TEST` y vemos que lista el contenido del directorio TEST





12) Para verificar el comando de cambio de permisos abrimos otra consola, nos dirigimos a la ubicación de la minishell y mediante el comando `ls -l` podemos ver todos los permisos de los archivos del directorio. Miramos el último... `TEST` que es el que vamos a modificarle los permisos durante esta prueba, inicialmente tiene los 3 permisos (`rw`x)

```
brendam@raspberrypi: ~/Desktop/Minishell
File Edit Tabs Help
brendam@raspberrypi:~ $ cd /home/brendam/Desktop/Minishell
brendam@raspberrypi:~/Desktop/Minishell $
brendam@raspberrypi:~/Desktop/Minishell $ ls -l
total 168
-rwxr-xr-x 1 brendam brendam 15520 Nov  4 11:05 ayuda
-rw-rw-rw- 1 brendam brendam 1021 Nov  4 09:50 ayuda.c
-rwxr-xr-x 1 brendam brendam 15640 Nov  4 11:05 crearArchivo
-rw-rw-rw- 1 brendam brendam 378 Nov  3 22:05 crearArchivo.c
-rwxr-xr-x 1 brendam brendam 15644 Nov  4 11:05 crearDirectorio
-rw-rw-rw- 1 brendam brendam 411 Nov  4 09:59 crearDirectorio.c
-rwxrwxrwx 1 brendam brendam 360 Nov  3 14:26 ejecutar.sh
-rw-r--r-- 1 brendam brendam 0 Nov  4 10:48 ejemplo1
-rwxr-xr-x 1 brendam brendam 15644 Nov  4 11:05 eliminarDirectorio
-rw-rw-rw- 1 brendam brendam 482 Nov  4 10:33 eliminarDirectorio.c
-rwxr-xr-x 1 brendam brendam 15752 Nov  4 11:05 listarContenidoDirectorio
-rw-rw-rw- 1 brendam brendam 544 Nov  3 22:19 listarContenidoDirectorio.c
-rwxr-xr-x 1 brendam brendam 15940 Nov  4 11:05 minishell
-rw-rw-rw- 1 brendam brendam 2333 Nov  3 14:32 minishell.c
-rwxr-xr-x 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
-rw-rw-rw- 1 brendam brendam 1394 Nov  4 09:52 modificarPermisos.c
-rwxr-xr-x 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoArchivo
-rw-rw-rw- 1 brendam brendam 537 Nov  3 22:38 mostrarContenidoArchivo.c
drwxr-xr-x 2 brendam brendam 4096 Nov  4 11:03 TEST
brendam@raspberrypi:~/Desktop/Minishell $
```

cambiamos el permiso de TEST/ejemplo2, sacando el de lectura, nos notifica que el permiso se modificó correctamente.

en la otra consola podemos ver como cambia el permiso del archivo...siendo este wx.

```
brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help

e indicado.
-chmod nombre permisos : Cambia los permisos indicados de
un archivo en el directorio con ese nombre.
Los comandos de los permisos son:
    Para escritura: '+escritura' 0 '-escritura'
    Para lectura: '+lectura' 0 '-lectura'
    Para ejecucion: '+ejecutar' 0 '-ejecutar'
-exit : Sale de la Minishell.

-> chmod TEST -lectura
permiso correctamente modificado
-> █

brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help

-rwxr-xr-x 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
-rw-rw-rw- 1 brendam brendam 1394 Nov  4 09:52 modificarPermisos
-rwxr-xr-x 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoA
-rw-rw-rw- 1 brendam brendam 537 Nov  3 22:38 mostrarContenidoA
d-wxr-xr-x 2 brendam brendam 4096 Nov  4 11:03 TEST
brendam@raspberry:~/Desktop/Minishell $ █
```

ingresando `chmod TEST/ejemplo2 +lectura` le damos permiso de lectura al archivo, nos avisa que se cambiaron correctamente los permisos y ahora podemos hacer un `cat` de `TEST/ejemplo2`.

```
-> chmod TEST/ejemplo2 +lectura
permiso correctamente modificado
-> cat TEST/ejemplo2
Esto es un archivo de texto
Para probar la minishell.
$0 2024.

-> █
```

En la otra consola podemos visualizar como se cambio efectivamente el permiso siendo este `rwx`

```
-> chmod TEST +lectura
permiso correctamente modificado
->

brendam@raspberrypi: ~/Desktop/Minishell

File Edit Tabs Help

-rwxr-xr-x 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
-rw-rw-rw- 1 brendam brendam  1394 Nov  4 09:52 modificarPermisos.c
-rwxr-xr-x 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoArchivo
-rw-rw-rw- 1 brendam brendam   537 Nov  3 22:38 mostrarContenidoArchivo.c
drwxr-xr-x 2 brendam brendam  4096 Nov  4 11:03 TEST
brendam@raspberrypi:~/Desktop/Minishell $
```

ingresando `chmod TEST -escritura`, sacamos el permiso de escritura al directorio TEST y vemos que al querer crear un archivo nos notifica que no tiene permiso

```
-> chmod TEST -escritura
permiso correctamente modificado
-> touch TEST/ejemploNuevo
Error al crear el Archivo: Permission denied
->
```

en la otra consola podemos ver como se quito el permiso de escritura, quedando asi

```
brendam@raspberrypi: ~/Desktop/Minishell

File Edit Tabs Help

Para escritura: '+escritura' 0 '-escritura'
Para lectura: '+lectura' 0 '-lectura'
Para ejecucion: '+ejecutar' 0 '-ejecutar'
-exit : Sale de la Minishell.

-> chmod TEST -lectura
permiso correctamente modificado
-> chmod TEST +lectura
permiso correctamente modificado
-> chmod TEST -escritura
permiso correctamente modificado
->

brendam@raspberrypi: ~/Desktop/Minishell

File Edit Tabs Help

-rwxr-xr-x 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
-rw-rw-rw- 1 brendam brendam  1394 Nov  4 09:52 modificarPermisos.c
-rwxr-xr-x 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoArchivo
-rw-rw-rw- 1 brendam brendam   537 Nov  3 22:38 mostrarContenidoArchivo.c
lr-xr-xr-x 2 brendam brendam  4096 Nov  4 11:03 TEST
brendam@raspberrypi:~/Desktop/Minishell $
```

Ingresando `chmod TEST +escritura` le damos permiso de escritura al directorio TEST, probamos de crear de un archivo y se crea correctamente.

```
brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help
Error al crear el Archivo: Permission denied

-> chmod TEST +escritura
permiso correctamente modificado
-> touch TEST/ejemploNuevo
Archivo creado correctamente

-> 
```

En la otra consola podemos ver el cambio de permisos

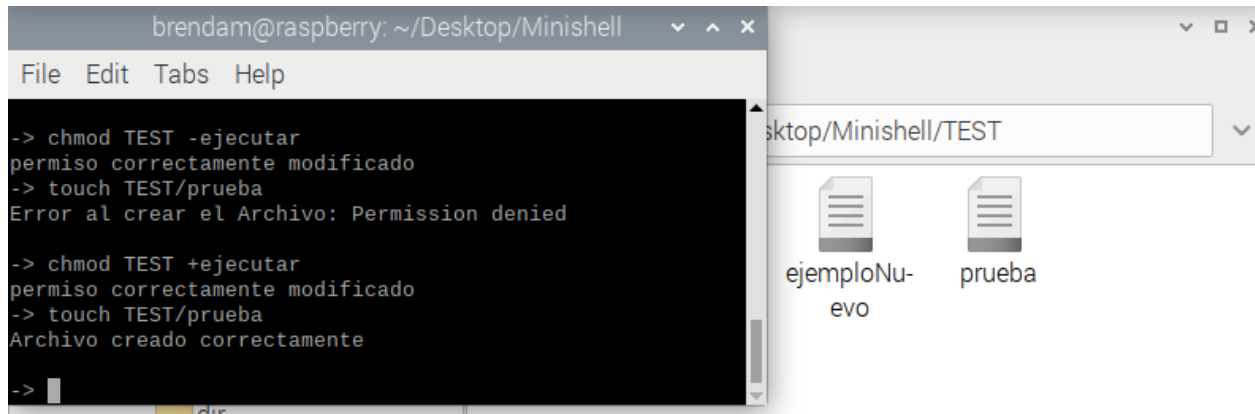
```
brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help
Para ejecucion: '+ejecutar'0'-ejecutar'
-exit : Sale de la Minishell.

-> chmod TEST -lectura
permiso correctamente modificado
-> chmod TEST +lectura
permiso correctamente modificado
-> chmod TEST -escritura
permiso correctamente modificado
-> chmod TEST +escritura
permiso correctamente modificado
-> 
```

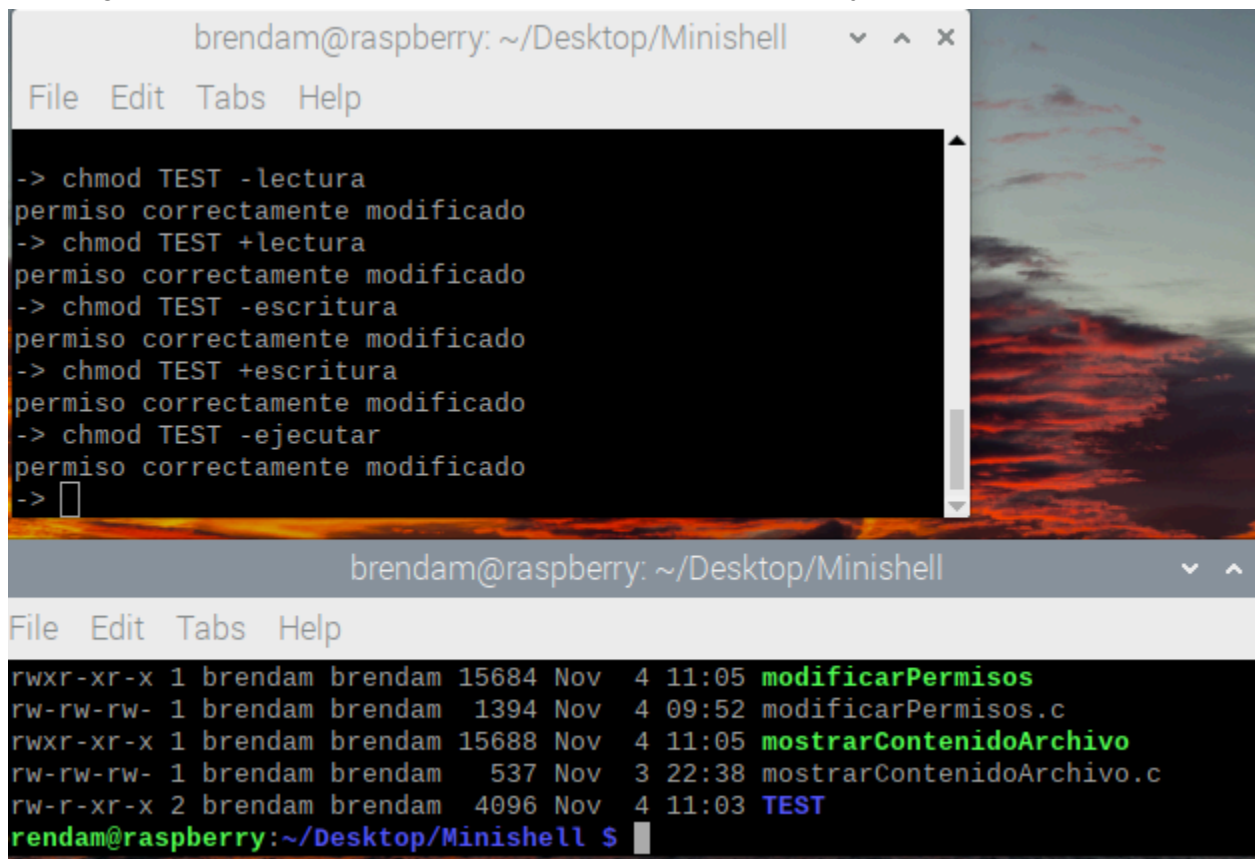
```
brendam@raspberry: ~/Desktop/Minishell
File Edit Tabs Help
rw-r--r-- 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
rw-rw-rw- 1 brendam brendam  1394 Nov  4 09:52 modificarPermisos.c
rw-r--r-- 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoArchivo
rw-rw-rw- 1 brendam brendam   537 Nov  3 22:38 mostrarContenidoArchivo.c
rw-r--r-- 2 brendam brendam  4096 Nov  4 11:03 TEST
brendam@raspberry:~/Desktop/Minishell $
```

Al ingresar el comando `chmod TEST -ejecutar` le cambiamos el permiso al directorio TEST y vemos que no se puede crear un archivo.

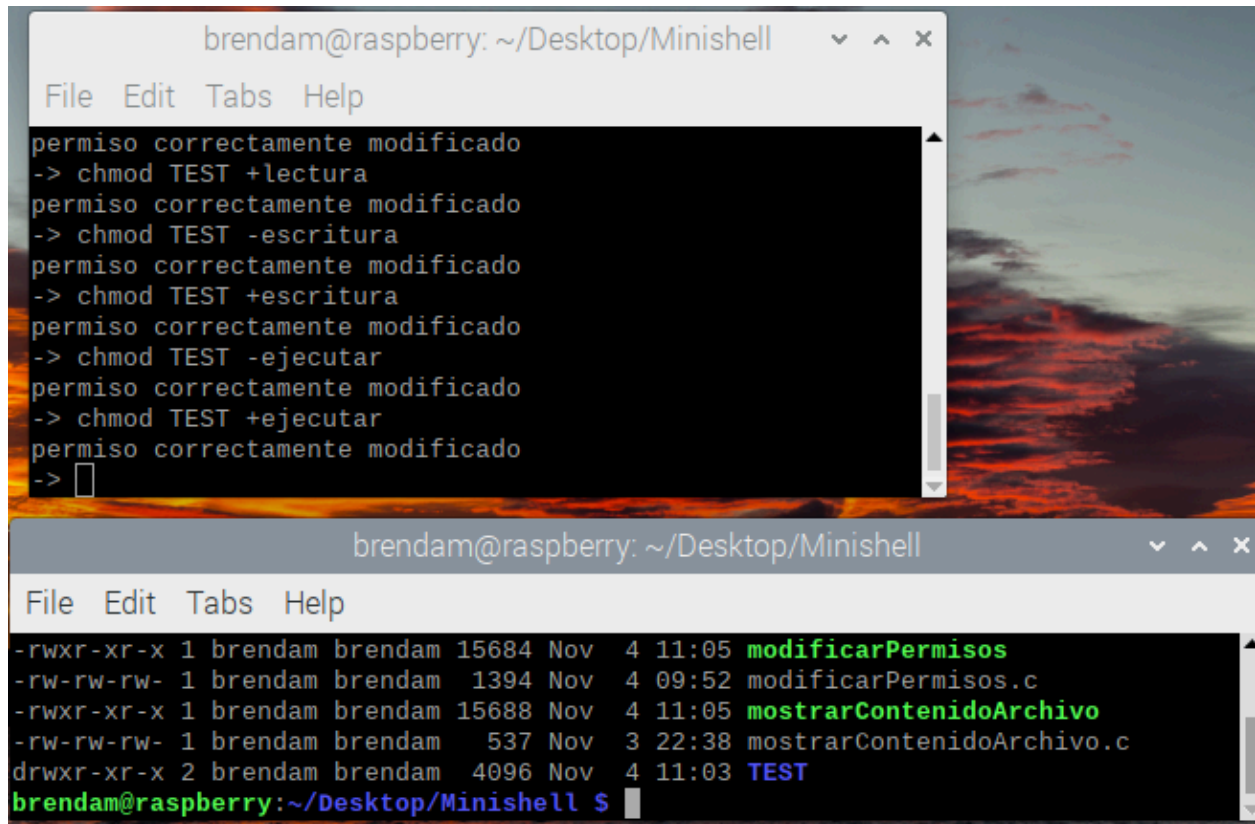
Al ingresar el comando `chmod TEST +ejecutar` le cambiamos el permiso al directorio TEST y podemos crear un archivo.



En la siguiente captura se ve como cambia el permiso al hacer -ejecutar



en la siguiente captura se ve como cambia el permiso al hacer +ejecutar

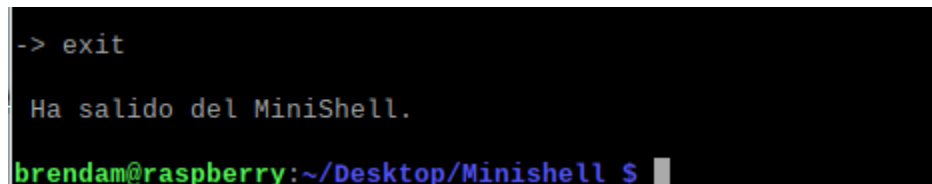


The image shows two overlapping terminal windows from a Raspberry Pi. The top window displays a series of commands to modify permissions for a file named 'TEST'. Each command is preceded by the text 'permiso correctamente modificado'. The bottom window shows the output of the 'ls' command, listing files with their permissions, owner, group, size, date, and name.

```
brendam@raspberrypi: ~/Desktop/Minishell
File Edit Tabs Help
permiso correctamente modificado
-> chmod TEST +lectura
permiso correctamente modificado
-> chmod TEST -escritura
permiso correctamente modificado
-> chmod TEST +escritura
permiso correctamente modificado
-> chmod TEST -ejecutar
permiso correctamente modificado
-> chmod TEST +ejecutar
permiso correctamente modificado
->

brendam@raspberrypi: ~/Desktop/Minishell
File Edit Tabs Help
-rwxr-xr-x 1 brendam brendam 15684 Nov  4 11:05 modificarPermisos
-rw-rw-rw- 1 brendam brendam  1394 Nov  4 09:52 modificarPermisos.c
-rwxr-xr-x 1 brendam brendam 15688 Nov  4 11:05 mostrarContenidoArchivo
-rw-rw-rw- 1 brendam brendam   537 Nov  3 22:38 mostrarContenidoArchivo.c
drwxr-xr-x 2 brendam brendam  4096 Nov  4 11:03 TEST
brendam@raspberrypi:~/Desktop/Minishell $
```

13) Por último ingresamos exit para salir de la mini shell y nos muestra el mensaje “ Ha salido del Minishell.



The image shows a terminal window where the 'exit' command has been entered. The output is 'Ha salido del MiniShell.' followed by the prompt 'brendam@raspberrypi:~/Desktop/Minishell \$'.

```
-> exit

Ha salido del MiniShell.
brendam@raspberrypi:~/Desktop/Minishell $
```

## 1.2. Sincronización

### 1. Taller de Motos.

Una pequeña fábrica de motos en la que trabajan seis operarios funciona de la siguiente manera. El primer operario arma ruedas de a una. Una vez que consigue armar dos, deja el lugar para el segundo operario que se encarga de armar el cuadro (chasis). El tercer operario es el encargado de agregar el motor al vehículo. Luego de esto la moto es pintada por uno de los dos pintores del taller. La moto puede ser pintada de verde o de rojo, según la elección del pintor que elija primero. En este punto la moto se encuentra lista, aunque una de cada dos motos es elegida para recibir un equipamiento extra. El sexto operario se ocupa de esta última tarea. Como el taller es pequeño, las tareas se realizan de a una y cada operario debe respetar su turno para trabajar, debiendo completar una moto entera antes de comenzar la siguiente.

Resuelva el problema utilizando hilos y semáforos.

Haga un uso eficiente de los recursos (hilos y semáforos).

Archivo con la resolución adjunta.

-Para resolver este problema se utilizaron 5 semaforos

- `ruedas_armadas`
- `chasis_listo`
- `motor_listo`
- `pintura_hecha`
- `equipamiento_listo`

-y un mutex: **`mutex_pintura`** para evitar que ambos operarios a cargo de la pintura, pinten la misma moto.

- 6 hilos, uno para cada operario `pthread_t operarios[6]`

- la inicializacion de los semaforos es `ruedas_armadas` en 0, `chasis_listo` en 0, `motor_listo` en 0, `pintura_hecha` en 0 y `equipamiento_listo` en 1 para empezar el ciclo.

-las funciones:

- **`operario_armar_ruedas()`**: decrementa el semáforo `equipamiento_listo`, es decir, espera que la moto este completa para empezar la siguiente. Se arman las dos ruedas y se incrementa el semáforo `ruedas_armadas` para dar aviso a que el operario de armar chasis puede comenzar a trabajar.
- **`operario_armar_chasis()`** : decrementa el semáforo `ruedas_armadas`, es decir, espera a que las ruedas estén listas el operario arma el chasis e incrementa el semáforo `chasis_listo` para indicar que el chasis está listo y el operario encargado del motor empiece a trabajar.



- **operario\_agregar\_motor()** : decrementa el valor del semáforo chasis\_listo, es decir, espera a que el chasis esté listo. Agrega el motor e incrementa el valor del semáforo motor\_listo para indicar que el motor está listo y el operario encargado de la pintura pueda empezar a trabajar.
- hay dos operarios a cargo de la pintura **operario\_pintar\_rojo()** y **operario\_pintar\_verde()** ambos decrementan el valor del semáforo motor\_listo esperando a que el motor esté listo y luego aquí se usa el mutex, para proteger la sección crítica de ambos operarios, una vez que uno de los dos accede a la sección crítica, pinta, libera el mutex y luego incrementa el valor del semáforo pintura\_hecha para indicar que la pintura esta lista.
- **operario\_agregar\_equipamiento()** : decrementa el valor del semaforo pintura\_hecha y decide de manera aleatoria si agregar equipamiento o no. Por último incrementa el valor del semáforo equipamiento\_listo para avisar que la moto esta lista.

## 2. Santa Claus.

Santa Claus duerme en su tienda en el Polo Norte y solo puede ser despertado por: los nueve renos, una vez que regresan de sus vacaciones en el Pacífico Sur algunos de los elfos que están teniendo dificultades para hacer juguetes. Para permitir que Santa descanse lo mejor posible, los elfos solo pueden despertarlo cuando tres de ellos tienen problemas. Cuando estos tres elfos están resolviendo sus problemas, cualquier otro elfo que quiera visitar a Santa debe esperar a que estos elfos terminen. En el momento que el último reno arriba al Polo Norte, este debe ir a buscar a Santa mientras los demás esperan en una cálida cabaña antes de ser enganchados al trineo. Si Santa se despierta y encuentra a tres elfos esperando en la puerta de su tienda, junto con el último reno que ha regresado de los trópicos, atiende primero a los renos, ya que es más importante preparar su trineo.

a) Resuelva este problema utilizando hilos (threads) y semáforos para su sincronización. Recuerde hacer uso eficiente de los recursos como por ejemplo la cantidad de semáforos que utiliza.

b) Explique brevemente el modelo implementado.

a)

Archivo adjunto con la resolución.

b) La resolución de este problema mediante el uso de hilos y semáforos está dada por:  
-5 semáforos en donde:

- **sem\_santa**: Santa se despierta cuando los renos o los elfos lo necesitan (semáforo binario).
- **sem\_nueveRenos**: Indica si los nueve renos han llegado (semáforo binario).
- **sem\_tresElfos**: Indica si tres elfos necesitan ayuda (semáforo binario).

- **sem\_renos y sem\_elfos**: Controlan la cantidad de renos y elfos que pueden recibir atención de Santa (semáforo de conteo).

## 1. Inicialización

- **sem\_santa = 0** (Santa dormido)
- **sem\_nueveRenos = 0** (ningún grupo de 9 renos listos)
- **sem\_tresElfos = 0** (ningún grupo de 3 elfos esperando ayuda)
- **sem\_renos = 9** (capacidad inicial para 9 renos)
- **sem\_elfos = 3** (capacidad inicial para 3 elfos)

-Dos mutex:

- **mutexRenos** : Mutex para asegurar que solo un hilo reno acceda a la sección crítica del código. función reno()
- **mutexElfos**: Mutex para asegurar que solo un solo hilo elfo acceda a su sección crítica del código. función elfo()).

-La función elfo() es la encargada de hacer que cuando 3 elfos necesiten ayuda, decrementar el semáforo sem\_elfos. Incrementar el semáforo sem Santa para despertar a santa e incrementar el semáforo sem tres Elfos para que santa atienda a los elfos.

-la función reno() es la encargada de que cuando renos llegan al polo norte, decrementar 9 veces el valor del semáforo sem\_renos. Incrementar el semáforo sem\_Santa para despertar a santa e incrementar el semaforo sem\_nueveRenos para que Santa atienda a los renos.

-la funcion santa() decrementa el valor del semaforo sem\_santa. luego con el sem\_trywait(&sem\_nueveRenos) intenta disminuir el conteo del semáforo sem\_nueveRenos. Este semáforo indica si los 9 renos han llegado. Si la llamada es exitosa (devuelve 0), significa que Santa puede atender a los renos,los atiende e incrementa el semaforo sem\_renos en 9. Luego con sem\_trywait(&sem\_tresElfos) intenta disminuir el conteo del semaforo tresElfos. Este semaforo indica si 3 elfos necesitan ayuda. Si la llamada es exitosa (devuelve 0), significa que santa puede ayudar a los elfos, los ayuda e incrementa el semáforo sem\_elfos en 3.

## 2. Problemas

### 2.1. Lectura

El 19 de julio de 2024, la empresa estadounidense de ciberseguridad CrowdStrike distribuyó una actualización defectuosa de su software de seguridad Falcon Sensor que causó problemas generalizados en las computadoras con Microsoft Windows que ejecutan el software. Como resultado, aproximadamente 8,5 millones de sistemas fallaron y no pudieron reiniciarse adecuadamente en lo que se ha denominado la mayor interrupción en la historia de la tecnología de la información. Se pide que investiguen uno de los temas mencionados abajo, relacionados con seguridad y protección, y generen alguna propuesta multimedia para visualizar el mismo. Puede realizarse en formato Flyer, Podcast, presentación o formato similar. Debe resaltar las características que considere más importantes. Deberán mostrar, de forma breve, su propuesta el día de la entrega del proyecto (de 3 a 5 minutos).

**Seguridad: Fallo de integridad** Presentación adjunta.

### 2.2. Problemas Conceptuales

#### 1. Paginación y Segmentación en Memoria

Escriba la traducción binaria de la dirección lógica (16 bits) 0011000000110011 bajo los siguientes esquemas hipotéticos de administración de memoria y explique su respuesta.

Muestre un diagrama donde se visualice cada uno de los esquemas mencionados:

- a) Un sistema de paginación con un tamaño de página de 512 direcciones, utilizando una tabla de páginas en la que el número de marco, en este caso, resulta ser la mitad del número de página. Es decir, como condición particular de este problema, si el número de página es  $P$ , el número de marco es  $M = P/2$
- b) Un sistema de segmentación con un tamaño máximo de segmento de 2K direcciones, utilizando una tabla de segmentos en la que las bases se colocan regularmente en direcciones reales:  $20 + 4,096 + \text{Nro Segmento}$ .

---

Dirección lógica (16 bits) 0011000000110011  
en decimal : 12339

### a) Número de página y desplazamiento:

- **Tamaño de página:** 512 direcciones.

Se divide a la dirección lógica según el número de bits correspondientes al

Número de página( P) y al Desplazamiento (D).

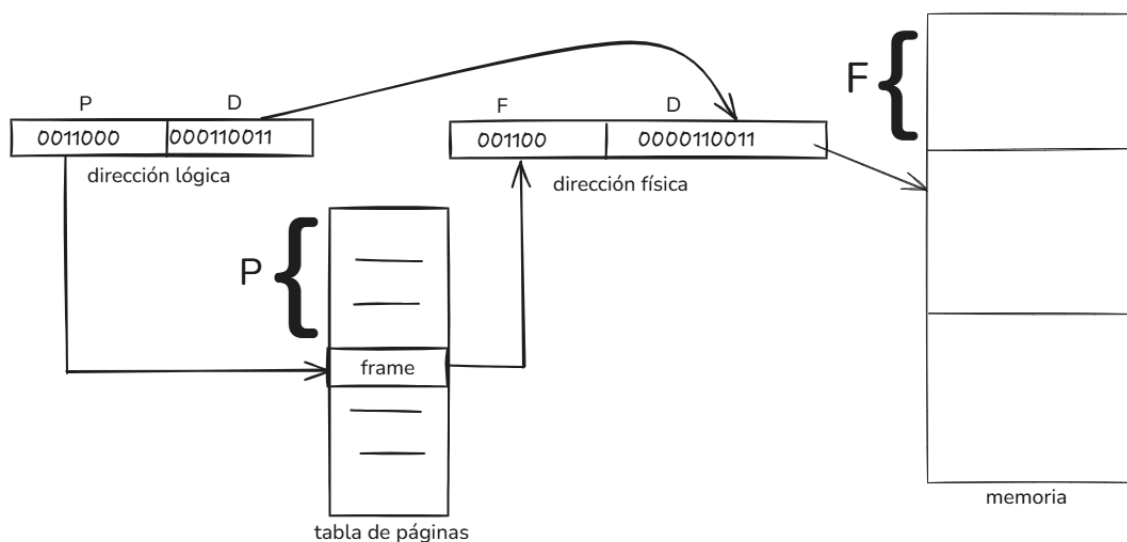
- Como  $512 = 2^9 \rightarrow$  **Desplazamiento (D)= 9**  $\rightarrow$  **000110011** (en binario), que equivale a **51** en decimal.
- $16 \text{ bits} - 9 \text{ bits} = 7 \text{ bits}$ , por lo que **Número de página( P)= 7 bits**  $\rightarrow$  **0011000** (en binario), que equivale a **24** en decimal.

### Número de marco:

- Por enunciado el número de marco(M) es la mitad del número de página( P)  $\rightarrow$   **$M = P/2$**
- Como  $P=24$ , el número de marco será  **$M=24/2=12$**

### Dirección física:

- Número de marco en binario (12 en decimal) es **0001100**
- El desplazamiento era **000110011** (9 bits).
- Luego la dirección física es: **0001100000110011** (en binario)



b)

**Tamaño máximo del segmento:** 2K direcciones.

- $2K=2048 \rightarrow 2^{11}$ , lo que significa que **11 bits** están destinados al desplazamiento.
- Como la dirección lógica tiene 16 bits.  $16-11 = 5$  bits que se utilizan para el número de segmento.

**Número de segmento y desplazamiento:**

- Para la dirección lógica **0011000000110011**:

Número de segmento (5 bits): **00110** (en binario), que sería segmento **6** en decimal.

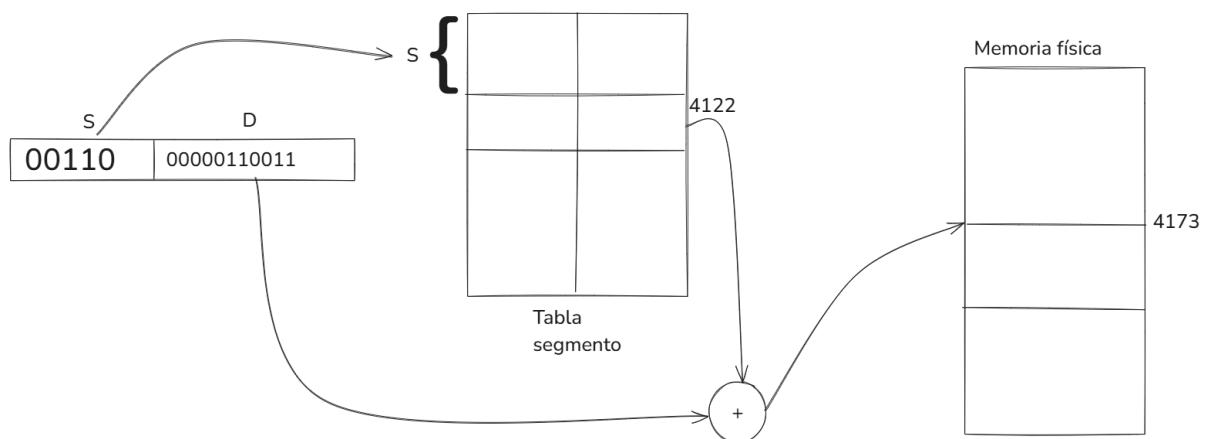
Desplazamiento (11 bits): **00000110011** (en binario), que sería desplazamiento **51** en decimal.

**Dirección base del segmento:**

- La base para cada segmento se coloca regularmente en direcciones reales, calculadas como:  $20 + 4,096 + \text{Nro Segmento}$
- Para el segmento 6: Base =  $20+4096+6=4122$ . y en binario: 1000000011010

**Dirección física:**

- La dirección física se obtiene sumando la base del segmento y el desplazamiento.
- Dirección física =  $4122+51=4173$ . y en binario: 1000001001101



## 2. Tabla de páginas

Considera la tabla de páginas para un sistema con direcciones virtuales y físicas de 16 bits y páginas de 4,096 bytes. El bit de referencia de una página se establece en 1 cuando la página ha sido referenciada. Periódicamente, un hilo pone a cero todos los valores del bit de referencia. Un guión en un marco de página indica que la página no está en memoria. El algoritmo de reemplazo de páginas es LRU con reemplazo local, y todos los números se proporcionan en decimal.

Pagina	Marco	Bit Referencia
0	9	0
1	-	0
2	10	0
3	15	0
4	6	0
5	13	0
6	8	0
7	12	0
8	7	0
9	-	0
10	5	0
11	4	0
12	1	0
13	0	0
14	-	0
15	2	0

a) Convierte las siguientes direcciones virtuales (en hexadecimal) a las direcciones físicas equivalentes. Puedes proporcionar las respuestas en hexadecimal o decimal. También establece el bit de referencia para la entrada correspondiente en la tabla de páginas.

- 0x621C
- 0xF0A3
- 0xBC1A

- 0x5BAA
- 0x0BA1

b) Usando las direcciones anteriores como guía, proporciona un ejemplo de una dirección lógica (en hexadecimal) que resulte en un fallo de página.

c) ¿De qué conjunto de marcos de página elegir a el algoritmo de reemplazo de páginas LRU al resolver un fallo de página?

---

Datos:

- Direcciones virtuales de 16 bits
- Direcciones físicas de 16 bits
- Páginas de 4096 bytes ( $2^{12}$ , por lo tanto, 12 bits para el offset).
- Algoritmo de reemplazo de páginas LRU

a)

### 0x621C

en binario : 0110001000011100

donde los primeros 4 bits corresponden al numero de pagina y los restantes 12 bits al desplazamiento teniendo así:

- Página: 0110 -> 6 en decimal.
- Desplazamiento: 001000011100 -> 540 en decimal.
- Para la página 6 el marco es 8 (1000) en memoria.
- Luego la dirección física es el marco+desplazamiento, quedando así:  
Dirección física: 1000001000011100 en binario y 0x821C en hexadecimal.
- Actualizar el bit de referencia a 1 a la página 6.

### 0xF0A3

en binario : 1111000010100011

donde los primeros 4 bits corresponden al numero de pagina y los restantes 12 bits al desplazamiento teniendo así:

- Página: 1111 -> 15 en decimal.
- Desplazamiento : 000010100011 -> 163 en decimal.
- Para la página 15 el marco es 2 (0010) en memoria.
- Luego la dirección física es el marco+desplazamiento, quedando así:  
Dirección física: 0010000010100011 en binario y 0x20A3 en hexadecimal.
- Actualizar el bit de referencia a 1 a la página 15.

### **0xBC1A**

en binario : 1011110000011010

donde los primeros 4 bits corresponden al numero de pagina y los restantes 12 bits al desplazamiento teniendo así:

- Página: 1011 -> 11 en decimal.
- Desplazamiento: 110000011010 -> 3098 en decimal.
- Para la página 11 el marco es 4 (0100) en memoria.
- Luego la dirección física es el marco+desplazamiento, quedando así:  
Dirección física: 0100110000011010 en binario y 0x4C1A en hexadecimal.
- Actualizar el bit de referencia a 1 a la página 11.

### **0x5BAA**

en binario : 0101101110101010

donde los primeros 4 bits corresponden al numero de pagina y los restantes 12 bits al desplazamiento teniendo así:

- Página: 0101 -> 5 en decimal.
- Desplazamiento: 101110101010 -> 2986 en decimal.
- Para la página 5 el marco es 13 (1101) en memoria.
- Luego la dirección física es el marco+desplazamiento, quedando así:  
Dirección física: 1101101110101010 en binario y 0xDBAA en hexadecimal.
- Actualizar el bit de referencia a 1 a la página 5.

### **0x0BA1**

en binario : 0000101110100001

donde los primeros 4 bits corresponden al numero de pagina y los restantes 12 bits al desplazamiento teniendo así:

- Página: 0000-> 0 en decimal
- Desplazamiento: 101110100001 -> 2977 en decimal.
- Para la página 0 el marco es 9 (1001) en memoria.
- Luego la dirección física es el marco+desplazamiento, quedando así:  
Dirección física: 1001101110100001 en binario y 0x9BA1 en hexadecimal.
- Actualizar el bit de referencia a 1 a la página 0.

La tabla quedaría de la siguiente manera:



Página	Marco	Bit de referencia
0	9	1
1	-	0
2	10	0
3	15	0
4	6	0
5	13	1
6	8	1
7	12	0
8	7	0
9	-	0
10	5	0
11	4	1
12	1	0
13	0	0
14	-	0
15	2	1

**b) Dirección lógica : 0x9001**

en binario es : 1001000000000001

y su página es 9 (1001) Como la página 9 no tiene un marco asignado en memoria hay un fallo de página.

**c) El algoritmo de reemplazo de páginas menos usadas recientemente (LRU) :** El algoritmo LRU selecciona el marco que no ha sido utilizado durante el período más largo. Cuando se produce un fallo de página (es decir, cuando se intenta acceder a una página que no está en memoria), se revisan todos los marcos ocupados y se determina cuál fue el menos utilizado recientemente. Para implementar el LRU por completo, es necesario mantener una lista enlazada de todas las páginas en memoria, con la página de uso más reciente en la parte frontal y la de uso menos reciente en la parte final.

entonces tenemos que los marcos con bit de referencia 0, es decir los que no han sido referenciados recientemente : Marcó 10 (Página 2)

Marco 15 (Página 3)

Marco 6 (Página 4)

Marco 12 (Página 7)

Marco 7 (Página 8)

Marco 5 (Página 10)

Marco 1 (Página 12)

Marco 0 (Página 13)

y los marcos con bit 1 tenemos

Marco 9 (Página 0)

Marco 13 (Página 5)

Marco 8 (Página 6)

Marco 4 (Página 11)

Marco 2 (Página 15)