



INSTITUTO POLITÉCNICO NACIONAL  
UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA Y  
CIENCIAS SOCIALES Y ADMINISTRATIVAS



# Práctica 4: Tuberías.

Yong Rodríguez Luz María

Programa 4.1: Crear una tubería entre padre e hijo que envía las letras del abecedario en mayúsculas. El hijo recibe y el padre envía. Los caracteres que se insertan están en su equivalente en código ASCII.

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
GNU nano 2.2.6 Archivo: programa4-1.c

#include<sys/types.h>
#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
int main ()
{
    int bytes_enviados, bytes_leidos, i, descriptor_archivo[2];
    char buffer[BUFSIZ + 1];
    pid_t pid;

    if(pipe(descriptor_archivo)==0){
        pid=fork();
        if(pid==-1){
            perror("Error al ejecutar fork");
            exit(-1);
        }
        if(pid==0){
            for(i=1;i<27;i++){
                [ 35 líneas leídas ]
            }
        }
    }
}
```

^G Ver ayuda ^O Guardar ^R Leer fich.^Y Pág. ant. ^K Cortar Tex ^C Posición  
^X Salir ^J Justificar ^W Buscar ^V Pág. sig. ^U PegarTxt ^T Ortografía

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
luzyong@luzyong-VirtualBox:~/sistemas_operativos/practica_4/src$ ./programa4-1
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: A
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: B
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: C
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: D
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: E
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: F
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: G
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: H
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: I
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: J
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: K
El proceso padre ha enviado por la tubería 1 bytes
```

- Se enviarían primero todos los datos y después imprimiría en pantalla los datos recibidos.

```
GNU nano 2.2.6 Archivo: pregunta5-1-1.c

if(pipe(descriptor_archivo)==0){
    pid=fork();
    if(pid==-1){
        perror("Error al ejecutar fork");
        exit(-1);
    }
    if(pid==0){
        for(i=1;i<27;i++){
            memset(buffer,'\0',sizeof(buffer));
            bytes_leidos = read(descriptor_archivo[0], buffer, BUFSIZ);
            sleep(1);
            printf("El proceso hijo ha leido %d bytes y el contenido es: %s\n", $
        )
        exit(0);
    }
}
else{
    for(i=65;i<91;i++){
        bytes_enviados=write(descriptor_archivo[1], (char *) &i, sizeof(ch$

^G Ver ayuda ^O Guardar ^R Leer fich. ^Y Pág. ant. ^K Cortar Tex ^C Posición
^X Salir ^J Justificar ^W Buscar ^V Pág. sig. ^U PegarTxt ^T Ortografía
```

```
luzyong@luzyong-VirtualBox: ~/sisistemas_operativos/practica_4/src
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
luzyong@luzyong-VirtualBox:~/sisistemas_operativos/practica_4/src$ El proceso hijo
ha leido 25 bytes y el contenido es: ABCDEFGHIJKLMNOPQRSTUVWXYZ
El proceso hijo ha leído 1 bytes y el contenido es: Z
```



¿Qué sucedería si no estuviese la función sleep(1)?

Pasaría lo mismo que en la pregunta anterior, con la diferencia de que los datos se dividen en dos al mostrarlos en pantalla.

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
GNU nano 2.2.6 Archivo: pregunta5-1-2.c

    bytes_leidos = read(descriptor_archivo[0], buffer, BUFSIZ);
    printf("El proceso hijo ha leído %d bytes y el contenido es: %s\n", $
    }
    exit(0);
}
else{
    for(i=65;i<91;i++){
        bytes_enviados=write(descriptor_archivo[1], (char *) &i, sizeof(ch$
        printf("El proceso padre ha enviado por la tubería %d bytes\n", by$
    }
}
}
exit(0);
}
```

^G Ver ayuda ^O Guardar ^R Leer fich.^Y Pág. ant. ^K Cortar Tex ^C Posición  
^X Salir ^J Justificar ^W Buscar ^V Pág. sig. ^U PegarTxt ^T Ortografía

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 14 bytes y el contenido es: ABCDEFGHIJKLMN
El proceso hijo ha leído 12 bytes y el contenido es: OPQRSTUVWXYZ
luzyong@luzyong-VirtualBox:~/sistemas_operativos/practica_4/src$
```

- Pregunta 4-2: ¿Qué sucedería si no existiese la función memset? ¿Y si solo existiese al inicio del código (antes de pipe)?

La función memset sirve para asignar un carácter determinado en los primeros n espacios de memoria. En el código se le asigna el valor \0 que es como limpiarlo para no causar conflictos. Por eso es que si lo ponemos antes del pipe o lo quitamos del código no pasa nada y el código sigue funcionando.

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
luzyong@luzyong-VirtualBox:~/sistemas_operativos/practica_4/src$ ./pregunta4-2-2
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: A
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: B
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: C
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: D
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: E
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: F
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: G
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: H
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: I
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: J
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: K
El proceso padre ha enviado por la tubería 1 bytes
```

1.-Código con memset antes de pipe

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
luzyong@luzyong-VirtualBox:~/sistemas_operativos/practica_4/src$ ./pregunta4-2-1
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: A
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: B
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: C
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: D
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: E
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: F
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: G
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: H
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: I
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: J
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: K
El proceso padre ha enviado por la tubería 1 bytes
```

2.-Código sin memset



Ejercicio 4-1: Modifique el programa 4-1 para determinar cuál es el tamaño de la tubería en su sistema.

Al principio del código declaramos una variable buffer con tamaño `BUSFIZ+1`. En la práctica nos menciona que los datos de la tubería se almacenan en un buffer con un tamaño determinado que depende de cada sistema. Para saber el tamaño de nuestra tubería imprimí el valor de `BUSFIZ`.

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
GNU nano 2.2.6 Archivo: ejercicio4-1.c

    bytes_leidos = read(descriptor_archivo[0], buffer, BUSFIZ);
    printf("El proceso hijo ha leído %d bytes y el contenido es: %s\n", $
    }
    exit(0);
}
else{
    for(i=65;i<91;i++){
        bytes_enviados=write(descriptor_archivo[1], (char *) &i, sizeof(ch$
        printf("El proceso padre ha enviado por la tubería %d bytes\n", by$
        sleep(1);
    }
}
printf("El tamaño de la tubería es %d\n", BUSFIZ);
exit(0);
}
```

**Ver ayuda** **Guardar** **Leer fich.** **Pág. ant.** **Cortar Tex** **Posición**  
**Salir** **Justificar** **Buscar** **Pág. sig.** **PegarTxt** **Ortografía**

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: P
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: Q
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: R
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: S
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: T
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: U
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: V
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: W
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: X
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: Y
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leído 1 bytes y el contenido es: Z
El tamaño de la tubería es 8192
luzyong@luzyong-VirtualBox:~/sistemas_operativos/practica_4/src$
```

Ejercicio 4-2: Modifique el programa 4-1 para que se añada una nueva tubería que pueda enviar datos del proceso hijo al proceso padre. El proceso hijo debe enviar las letras del abecedario, pero en minúsculas. En este código hice otra tubería después de la tubería con mayúsculas, en la que si el valor del pid era igual al proceso hijo, mandara el abecedario en minúsculas por medio del código ASCII.

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
GNU nano 2.2.6 Archivo: ejercicio4-2.c

    }
    }
}
//Creando nueva tubería
if(pipe(descriptor_archivo)==0){
    pid=fork();
    if(pid==-1){
        perror("Error al ejecutar fork");
        exit(-1);
    }
    if(pid==0){ //El proceso hijo le manda datos al padre
        for(i=97;i<123;i++){//La i es el valor de cada letra en código ASCII
            bytes_enviados=write(descriptor_archivo[1],(char *)&i, sizeof(char));
            printf("El proceso hijo ha enviado por la tubería %d bytes\n", $);
            sleep(1);
        }
    }
    else{
        for(i=1;i<27;i++){
            ^G Ver ayuda ^O Guardar ^R Leer fich.^Y Pág. ant. ^K Cortar Tex^C Posición
            ^X Salir ^J Justificar^W Buscar ^V Pág. sig. ^U PegarTxt ^T Ortografía
        }
    }
}
```

```
luzyong@luzyong-VirtualBox: ~/sistemas_operativos/practica_4/src
El proceso hijo ha leido 1 bytes y el contenido es: T
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: U
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: V
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: W
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: X
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: Y
El proceso padre ha enviado por la tubería 1 bytes
El proceso hijo ha leido 1 bytes y el contenido es: Z
El proceso hijo ha enviado por la tubería 1 bytes
El proceso padre ha leido 1 bytes y el contenido es: a
El proceso hijo ha enviado por la tubería 1 bytes
El proceso padre ha leido 1 bytes y el contenido es: b
El proceso hijo ha enviado por la tubería 1 bytes
El proceso padre ha leido 1 bytes y el contenido es: c
El proceso hijo ha enviado por la tubería 1 bytes
El proceso padre ha leido 1 bytes y el contenido es: d
El proceso padre ha leido 1 bytes y el contenido es: e
El proceso hijo ha enviado por la tubería 1 bytes
El proceso padre ha leido 1 bytes y el contenido es: f
```

### Conclusiones.

En esta práctica aprendí la función de pipe, cómo se escriben y se leen datos desde archivos y en espacios de memoria y cómo podemos enviar datos entre procesos por medio de las tuberías. Comprendí mejor el concepto de tubería y su funcionamiento.