Començat el	
Estat	Acabat
Completat el	
Temps emprat	
Punts	
Qualificació	

Pregunta **1**

Dado el siguiente trozo de código:

```
int fifo;
while((fifo = open("mipipe", O_WRONLY | O_NONBLOCK)) == -1 && errno == ENXIO && i<30)
{
        sleep(2);
        i++;
}
```

Asumiendo que la pipe con nombre "mipipe" existe, indica de las opciones siguientes, la que describe qué realiza este código:

Trieu-ne una:

- a. Intenta abrir una pipe con nombre "mipipe" para sólo escritura esperando a que aparezcan lectores durante al menos un minuto. ✔
- b. Espera por una señal SIGPIPE durante al menos 1 minuto.
- c. Intenta sobreescribir una pipe con nombre "mipipe" durante al menos 1 minuto.
- d. Intenta desbloquear una pipe con nombre "mipipe" durante al menos 1 minuto.

La resposta correcta és: Intenta abrir una pipe con nombre "mipipe" para sólo escritura esperando a que aparezcan lectores durante al menos un minuto.

Pregunta **2**

Dado el siguiente código incompleto:

```
int main() {
      char buf[256];
      int pd[2];
      pipe(pd);
      int pid = fork();
      if(pid == 0) {
              dup2(pd[1], 1);
              close(pd[0]);
              close(pd[1]);
              execlp("ls","ls",NULL);
      }
      else {
              close(pd[1]);
              read(<<x1>>, buf, sizeof(buf));
              write(<<x2>>, <<x3>>, <<x4>>);
              close(pd[0]);
              waitpid(-1,NULL,0);
      }
}
```

Indica cuál de las siguientes opciones son **CIERTAS** de forma que el resultado de la ejecución fuera la de la ejecución del comando *ls* (asumiendo que no se redirecciona la entrada ni la salida de la ejecución del programa).

Trieu-ne una:

- a. <<x1>> debería ser pd[0], <<x2>> debería ser pd[1], <<x3>> debería ser *buf* y <<x4>> debería ser *strlen(buf)*
- b. <<x1>> debería ser pd[0], <<x2>> debería ser 1, <<x3>> debería ser *buf* y <<x4>> debería ser *strlen(buf)* ✔
- c. <<x1>> debería ser pd[1], <<x2>> debería ser 0, <<x3>> debería ser *buf* y <<x4>> debería ser *sizeof(buf)*
- d. <<x1>> debería ser pd[0], <<x2>> debería ser pd[1], <<x3>> debería ser *buf* y <<x4>> debería ser *strlen(buf)*

La resposta correcta és: <<x1>> debería ser pd[0], <<x2>> debería ser 1, <<x3>> debería ser *buf* y <<x4>> debería ser *strlen(buf)*

Pregunta **3**

Dada la siguiente secuencia de comandos y salida asociada, ejecutada desde un terminal (**Terminal 1)**:

```
# mknod mipipe p
# ls
mipipe out.txt code.c
# ls > mipipe
```

En otra terminal (**Terminal 2**) en la misma máquina, y desde la misma carpeta que en la terminal anterior y ejecutamos el siguiente comando:

```
# cat < mipipe
```

De las siguientes opciones, indique cuál es la que muestra el resultado de la ejecución del comando en **Terminal 2:**

Trieu-ne una:

- a. En Terminal 2, se obtendrá la siguiente salida:
  code.c
  out.txt
- b. En Terminal 2, se obtendrá la siguiente salida:
  code.c
  mipipe
  out.txt
- c. En Terminal 2 el comando se queda en espera de que en Terminal 1 se escriban caracteres desde línea de comandos.
- d. En Terminal 2, el comando no muestra ningún resultado y se queda bloqueado ya que se no se indica fichero para mostrar.

La resposta correcta és: En Terminal 2, se obtendrá la siguiente salida:
code.c
mipipe
out.txt

Pregunta **4**

Una comunicación entre dos procesos que sea bidireccional, es decir que ambos procesos puedan tanto escribir como leer mensajes, se podría implementar de varias formas. Indica cuál de las siguientes afirmaciones es **CIERTA**:

Trieu-ne una o més:

- a. Con una única pipe con nombre sería suficiente ya que tiene dos extremos.
- b. Necesitamos dos pipes con nombre ya que ambos procesos leyendo/escribiendo una sola pipe se podría perder información o alguno de los procesos podría bloquearse. ✔
- c. Con una única pipe sin nombre sería suficiente ya que tiene dos extremos.
- d. Necesitamos dos pipes sin nombre ya que ambos procesos leyendo/escribiendo una sola pipe se podría perder información o alguno de los procesos podría bloquearse.
- e. Es imposible porque no comparten memoria.

Les respostes correctes són: Necesitamos dos pipes sin nombre ya que ambos procesos leyendo/escribiendo una sola pipe se podría perder información o alguno de los procesos podría bloquearse., Necesitamos dos pipes con nombre ya que ambos procesos leyendo/escribiendo una sola pipe se podría perder información o alguno de los procesos podría bloquearse.

Pregunta **5**

Dado el siguiente código:

```
int main() {
     int fd[2], ret;
     char buf[32];
     pipe (fd);
     if (fork() == 0) {
             close (fd[1]);
             ret = read(fd[0], buf, sizeof(buf));
             close(fd[0]);
             write (1, buf, ret);
     } else {
             close (fd[0]);
             sprintf(buf, "Toc 1\n");
             write (fd[1], buf, strlen(buf));
             sprintf(buf, "Toc 2\n");
             write (fd[1], buf, strlen(buf));
             sprintf(buf, "Toc 3\n");
             write (fd[1], buf, strlen(buf));
             close (fd[1]);
     }
```

Si ejecutamos el programa sin redireccionar la entrada ni la salida, indica de las opciones siguientes cuáles son **CIERTAS**:

Trieu-ne una:

- a. Por salida estándar se mostrará el mensaje "Toc 1". El proceso hijo acabará su ejecución pero el proceso padre quedará bloqueado esperando lectores.
- b. Por salida estándar no mostrará ningún mensaje porque se cierran los canales antes de poder leer ningún mensaje.
- c. Por salida estándar mostrará como mínimo los mensajes: "Toc 1", "Toc 2". Los procesos acabaran su ejecución.
- d. Por salida estándar mostrará como mínimo el mensaje "Toc 1", el resto de los mensajes podrían aparecen (en orden) pero no necesariamente. Los dos procesos acabaran su ejecución. ✔

La resposta correcta és: Por salida estándar mostrará como mínimo el mensaje "Toc 1", el resto de los mensajes podrían aparecen (en orden) pero no necesariamente. Los dos procesos acabaran su ejecución.

◄ Session 7 test

Salta a...

Session 8 test ►