Problema 1 (3,5 puntos).
Hemos escrito el siguiente programa en UNIX:
`programa.c:`

```
#include <signal.h>

void sig_alrm()
{
        write( 2, "RUTINA\n", 7 );
        return;
}

main()
{
 int fd[2], n;
 char mensaje[8], *s;

        s="MENSAJE\n";
        signal(SIGALRM,sig_alrm);
        pipe(fd);
        if( fork() == 0 ){
                close(fd[1]);
                alarm(3);
                while ((n = read( fd[0], mensaje, 8)) > 0 );
                alarm(0);
                exit(0);
        };
        close(1);
        dup(fd[1]);
        close(fd[0]);
        close(fd[1]);
        while(1)        write( 1, s, 8 );
}
```

Después de compilar "`programa.c`" y supuesto ya obtenido el ejecutable "`programa`" responde a las siguientes preguntas:

- Explica de forma detallada cuál será el resultado de la ejecución de "`programa`" y por qué.
- Explica qué efecto tiene en cualquier programa escrito para UNIX la línea `alarm(0);`
- Existe alguna diferencia en la ejecución anterior si a "`programa.c`" le quitamos la línea `alarm(0);` de su código?.

Problema 2 (3,5 puntos).

Diseñar un programa llamado "`crea`" en lenguaje C y usando sólo llamadas al sistema UNIX que actue como sigue:

- "`crea`" debe crear tantos hijos como se le indique en el primer parámetro pasado a "`crea`".
- cada uno de estos hijos deberá enviar a su padre su identificador de proceso (`PID`) mediante una pipe y a continuación morirá.
- el padre deberá escribir en un fichero (cuyo nombre es pasado como segundo parámetro a "`crea`") todos los mensajes que le lleguen de su hijo (uno en cada línea).

Nota: Todos los hijos deberan utilizar la misma pipe.

Ejemplo: **merlin_$** `crea 15 hijos`

Se habrán creado 15 hijos cuyos identificadores (`PID's`) aparecerán en el fichero "`hijos`".

Problema 3 (1 punto).

Recuerda cómo funciona el algoritmo de planificación de la CPU: "`primero el trabajo más breve`" (SJF, Shortest-Job-First) en su alternativa `no apropiativa` y calcula el tiempo promedio de espera que surge de la siguiente situación en la cola de Procesos Preparados:

Proceso	Instante de llegada	Duración de la ráfaga
P1	0	8
P2	1	4
P3	2	9
P4	3	5

Nota: Detalla tus cálculos lo mejor que puedas (ayudate de gráficas explicativas).

Problema 4 (2 puntos).

Al ejecutar el comando "`ls -li`" en un sistema UNIX aparece la siguiente salida en pantalla:

```
merlin_$ ls -li
total 1
54686 -rw-r--r--  2   user1   compu  0    Jun 19 10:21 fichero1
54686 -rw-r--r--  2   user1   compu  0    Jun 19 10:21 fichero2
```

La primera columna aparece como respuesta a la opción "`-i`" del comando, e indica el número de nodo-i de cada uno de los ficheros que contiene el directorio. El resto de columnas ya las conoceis, son las usuales de la opción "`-l`".

1. Suponiendo que cada uno de los siguientes comandos se ejecuta partiendo de la situación inicial. Indica qué salida produciría "`ls -li`" después de ejecutar cada uno de ellos?.

```
merlin_$ rm fichero1
merlin_$ chmod u+x fichero1
merlin_$ mv fichero1 ..
merlin_$ cp fichero1 fichero3
```