Ejercicio 1 (3 pts): Tras finalizar el diseño de nuestro sistema operativo

a) (1pt) Describa …

```
typedef struct sem_t {
        mutex_t *cerrojo;            // mutex
        cond_t  *cond;               // variable condicional
        int valor;                   // valor del semáforo
} semaforo;
```

b) (1pt) Implemente …

```
semaforo *crear_semaforo (int valor_inicial) {
        if (valor_inicial < 0)
                return NULL;
        semaforo *sem = malloc(sizeof(semaforo));
        pthrea
        d_mutex_init(sem->cerrojo,NULL);
        pthread_cond_init(sem->cond,NULL);
        sem->valor = valor_inicial;
        return sem;
}
```

c) (1pt) Finalmente …

```
void post(semaforo sem) {
        pthread_mutex_lock(sem->cerrojo);
        if (sem->valor == 0)
                pthread_cond_signal(sem->cond, sem->cerrojo);
        sem->valor++;
        pthread_mutex_unlock(sem->cerrojo);
}
void wait(semaforo sem) {
        pthread_mutex_lock(sem->cerrojo);
        while (sem->valor == 0)
                pthread_cond_wait(sem->cond, sem->cerrojo);
        sem->valor--;
        pthread_mutex_unlock(sem->cerrojo);
}
```

Ejercicio 2 (1.5 pts): Considerar la siguiente secuencia…

0x10, 0x1A, 0x1F4, 0x17C, 0x7C, 0x3B9, 0x185, 0x2FF, 0x24C, 0x434, 0x458, 0x36D

a) (0.5pts) Deducir la cadena de referencias …

La cadena de referencias es **0, 0, 1, 1, 0, 3, 1, 2, 2, 4, 4, 3**
y reducida **0, 1, 0, 3, 1, 2, 4, 3**

b) (1pt) Determinar razonadamente …

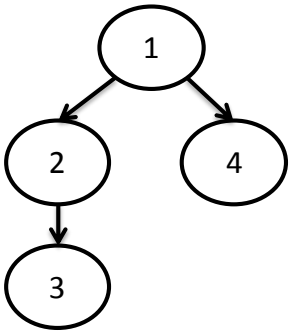
OPT	Marco 0	0 0 0 0 0 3 3 3 3 3 3	**5 fallos**
	1	1 1 1 1 1 2 2 4 4 4	
FIFO	Marco 0	0 0 0 0 0 3 3 3 4 4 4	**6 fallos**
	1	1 1 1 1 1 2 2 2 2 3	
LRU	Marco 0	0 0 0 0 0 0 1 1 1 4 4 4	**7 fallos**
	1	1 1 1 3 3 2 2 2 2 3	
Reloj	Marco 0	0 0'0'0'0'3 3 2 2'2'2'3	**6 fallos**
	1	1 1'1'1 1'1 1 4 4'4	

Ejercicio 3 (1.5 pt): Dado el siguiente código …

a) (0.5pts) Escriba la salida por pantalla del programa.

```
PID: 3, (2)
PID: 2, (1)
PID: 4, (1)
PID: 1, (0)
```

b) (0.5pts) ¿Qué esquema jerárquico de procesos genera este programa?



c) (0.5pts) ¿Cuántos procesos …

Tres, el 3, 2, y 1

Ejercicio 4 (1.5 pt): Supóngase un disco de 256 cilindros, 4 cabezas, 100 sectores …

a) (0.5pts) Determinar el tiempo de posicionamiento …

256 cilindros, 4 cabezas, tiempo de desplazamiento 0.5ms por cilindro.
CPS (25, 4, 12) -> CPS=(15,2,15)
-El disco gira a 6000rpm -> 60[s/min]/6000[rev/min] · 1000[ms/s]-> 10ms/rev
-Un sector tarda en leerse 10/100=0.1ms/sector
-Nos movemos 10 cilindros -> 0.5·10=5ms, lo que supone avanzar 50 sectores (media pista), así que estaremos en el 12+50=62, si queremos ir al 15 son 38 hasta el 0 mas 15 lo que hace 53 sectores -> 5ms + 5.3ms = 10.3ms

b) (0.5pts) Calcular …

100 sectores de 2KB por pista -> 200KB por pista, 800KB por cilindro
Si queremos leer 900KB, son 1 cilindro más 100KB, o sea, media pista más -> CHS=(16,2,65)
El tiempo de lectura son 10ms·(4+0.5) de lectura de datos, más un cambio de cilindro, pero en el cambio de cilindro dejamos de estar en la posición 0, así que hay que esperar a la vuelta siguiente.
Total=10ms·(4+0.5)+10ms=50.5ms.

c) (0.5pts) y la posición CPS de la cabeza tras la lectura.

CHS=(16,2,65)

Ejercicio 5 (2.5 pts): Un sistema de ficheros basado en i-nodos y mapa de bits contiene la siguiente información:

a) (1.5pts) Rellene los huecos para que el sistema sea consistente…

Mapa de bits: 1 0 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0 0 ……. 0

i-nodo 2

Tamaño	1
#Enlaces	NA
Tipo F/D	D
Directo	3
Indirecto	Null

i-nodo 3

Tamaño	2
#Enlaces	1
Tipo F/D	F
Directo	6
Indirecto	7

i-nodo 4

Tamaño	1
#Enlaces	2
Tipo F/D	F
Directo	12
Indirecto	Null

i-nodo 5

Tamaño	1
#Enlaces	NA
Tipo F/D	D
Directo	0
Indirecto	Null

i-nodo 9

Tamaño	1
#Enlaces	NA
Tipo F/D	D
Directo	5
Indirecto	Null

Bloque 0

.	5
..	2
C	9
D	4

Bloque 3

.	2
..	2
A	3
B	5
E	4

Bloque 5

.	9
..	5

Bloque 6: Datos sin formato

Bloque 7: 15

Bloque 12: Datos sin formato

Bloque 15: Datos sin formato

b) (1pt) Dibuje el árbol del directorio empleando óvalos para los directorios …