Problema 1 (3.25 puntos, 0.75 puntos por pregunta a menos que se indique lo contrario)

A continuación se muestra un código en C que compila y funciona correctamente.

```
 1 int main(void)
 2 {
 3   int ret;
 4   char *argv[3] = {"/usr/bin/grep",  "allow", NULL};
 5
 6   ret = fork();
 7
 8   if (ret == 0) {
 9     printf("Executing %s\n", argv[0]);
10     int fd1 = open("file1.txt", O_RDONLY);
11     int fd2 = open("file2.txt", O_WRONLY | O_CREAT, S_IRUSR | S_IWUSR);
12     dup2(fd1, 0);
13     dup2(fd2, 1);
14     close(fd1);
15     close(fd2);
16     execv(argv[0], argv);
17   } else {
18     printf("Waiting...\n");
19     wait(NULL);
20     printf("Finished.\n");
21     return 0;
22   }
23 }
```

Al ejecutar el código anterior se muestran los siguientes mensajes por pantalla.

```
Waiting...
Executing /usr/bin/grep
Finished.
```

En los siguientes apartados analizaremos paso por paso lo que hace el código. Leer primero todo el enunciado antes de comenzar a responder.

Para responder a las preguntas se pueden ignorar las llamadas a la función "close" que hay en el código. En caso que se prefiera no ignorarlos, indicarlo a la hora de responder. En caso de se haga algún otro supuesto indicarlo en la respuesta.

a) ¿Qué partes del código son las que se ejecutan primero y cuáles después? Razonar la respuesta basándose en los mensajes que se imprimen por pantalla.

b) Comenta brevemente qué es lo que hacen las líneas 10 y 11 del código. ¿Qué es lo que se almacenará en las variables "fd1" y "fd2"?

c) (0.5 puntos) Comentar brevemente qué es lo que hace la línea 12 del código. ¿Cuál es el objetivo de ejecutar esta línea?

d) (0.5 puntos) Comentar brevemente qué es lo que hace la línea 13 del código. ¿Cuál es el objetivo de ejecutar esta línea?

e) En la línea 16 se ejecuta una aplicación. Dadas las instrucciones C ejecutadas anteriores a esta línea, ¿qué es lo que hace el código? Para ayudarte a responder a la pregunta puedes indicar cuál es el comando equivalente a ejecutar en el terminal.

Problema 2 (3.00 puntos, 0.75 puntos por pregunta)

A continuación se muestra un código en C que compila y funciona correctamente.

```
1 #define N 10
2
3 int main()
4 {
5   pid_t pid;
6   int i, value, sum, fd[2];
7
8   pipe(fd);
9
10  i = N;
11  sum = 0;
12
13  for(i = 0; i < N; i++) {
14    pid = fork();
15    if (pid == 0) {
16      value = i;
17      printf("Value %d\n", value);
18      write(fd[1], &value, sizeof(int));
19      return 0;
20    }
21  }
22
23  for(i = 0; i < N; i++) {
24    read(fd[0], &value, sizeof(int));
25    sum += value;
26  }
27
28  printf("Value of sum: %d\n", sum);
29 }
```

En una de las ejecuciones del código anterior se muestran los siguientes mensajes por pantalla.

```
Value 0
Value 4
Value 6
Value 1
Value 8
Value 5
Value 2
Value 3
Value 7
Value 9
Value of sum: 45
```

A continuación analizaremos lo que hace el código. Responder a las siguientes preguntas de forma breve y concisa.

a) Describir la estructura de padre-hijos que se genera al ejecutar el código.

b) Dibuja y comenta el esquema de conexiones de comunicación que se genera entre los procesos.

c) ¿Qué proceso es el que ejecuta las líneas de código de 23 a 26? ¿El padre? ¿Algún hijo? Razonar la respuesta.

d) Comentar brevemente qué es lo que hace el código. En particular, comentar los mensajes que se envían por la tubería (quién los escribe, quién los lee) así como por qué el mensaje "Value of sum: 45" se imprime siempre al final de la ejecución del código aunque no haya ningún "wait" en el código.

Problema 3 (3.75 puntos en total, 0.75 puntos por pregunta) Comentar las siguientes afirmaciones de los sistemas operativos. En caso de que la frase sea incorrecta, indicar cómo funciona realmente y razonar la respuesta poniendo ejemplos si se cree conveniente. En caso que sea correcta, comentar la razón por la cual es correcta.

a) El sistema operativo es un proceso de usuario.

b) En caso de que un proceso ejecute una instrucción no autorizada se produce una llamada a sistema de forma que el sistema operativo toma el control y, en caso necesario, "mata"el proceso.

c) El núcleo de un sistema operativo se encarga de gestionar a qué partes de la memoria RAM pueden acceder los procesos.

d) Las interrupciones se producen al ejecutar una instrucción del proceso y hacen que el hardware cambie de modo usuario a modo núcleo.

e) Las llamadas a sistema son las funciones de las cuales disponen los procesos de usuario para acceder al hardware.