Apellidos:_____Nombre:_____ DNI: _____

Sistemas Operativos – Grado Ingeniería Informática UDC. Julio 2024
Sólo puede usar lápiz, bolígrafo y calculadora.

Parte **Sistema Ficheros (**2 puntos)

Examen sin identificación completa NO se califica.
Esta parte NO admite entrega de hojas adicionales. Hay que contestar en los espacios proporcionados.

Puntuación preguntas: P1: 0.6, P2: 0.3, P3: 0.5, P4:0.6

P1) Un sistema de archivos tipo system V tiene un tamaño de bloque de 2 Kbytes e inodos con 10 direcciones directas de bloques, una indirecta simple, una indirecta doble y una indirecta triple. Además, utiliza direcciones de bloques de 8 bytes. Consideremos un fichero con tamaño de 130 MBytes + 19 Kbytes.

Calcular cuántos bloques de disco son necesarios, en el área de datos, para representar ese archivo.

Número bloques de datos: ____ **65Kbloques+10**
Número de bloques de índices: ____ **263**
Fragmentación interna último bloque de datos: ____ **1Kbyte**

Un sistema de ficheiros tipo system V ten un tamaño de bloque de 2 Kbytes e inodos con 10 enderezos de bloque directos, un indirecto simple, un indirecto dobre e un indirecto triplo. Ademais, usa enderezos de bloque de 8 bytes. Consideremos un ficheiro cun tamaño de 130 MBytes + 19 Kbytes.

Calcula cantos bloques de disco son necesarios, na área de datos, para representar ese ficheiro.

Número de bloques de datos: ____
Número de bloques de índices: ____
Fragmentación interna último bloque de datos: ____

A system V file system has a block size of 2 Kbytes and inodes with 10 direct block addresses, one single indirection, one double indirection and one triple indirection. In addition, it uses 8-byte block addresses. Consider a file with size 130 MBytes + 19 Kbytes.

Calculate how many disk blocks are needed, in the data area, to represent that file.

Number of data blocks: ____
Number of index blocks: ____
Internal fragmentation last data block: ____

P2) Calcular qué número de bloque lógico corresponde al inodo del directorio raíz (se comienza a contar en el bloque lógico 0), y cuál bloque lógico al inodo de un fichero "datos", suponiendo lo siguiente:

i) El nº de inodo del "/" es el 2, y el fichero "datos" corresponde al inodo número 642 (los inodos se comienzan a numerar a partir de 1).
ii) El tamaño de un inodo es de 64 bytes (tamaño bloque = 2Kbytes).
iii) El boot ocupa 2 bloques y el superbloque 14 bloques.

Nº bloque lógico del inodo de "/":____ **16**

Nº bloque lógico del inodo de fichero "datos": ____ **36**

Calcular que número de bloque lóxico corresponde ao inodo do directorio raíz (o reconto comeza no bloque lóxico 0) e que bloque lóxico corresponde ao inodo dun ficheiro "datos", asumindo o seguinte:

i) O número de inodo do "/" é 2, e o ficheiro "datos" corresponde ao inodo número 642 (os inodos comezan a numerarse a partir de 1).
ii) O tamaño dun inodo é de 64 bytes (tamaño do bloque = 2Kbytes).
iii) O boot ocupa 2 bloques e o superbloque 14 bloques.

Número de bloque lóxico do inodo de "/":____

Número de bloque lóxico do inodo do ficheiro "datos": ____

Calculate which logical block number corresponds to the inode of the directory root (counting starts at logical block 0), and which logical block corresponds to the inode of a file "data", assuming the following:

i) The inode number of "/" is 2, and file "data" corresponds to inode number 642 (the inodes start numbering from 1).
ii) The size of an inode is 64 bytes (block size = 2Kbytes).
iii) The boot occupies 2 blocks and the superblock occupies 14 blocks.

Logical block number of the "/" inode:____

Logical block number of the inode of the file "data": ____

P3) En el sistema de archivos del problema P1 (tamaño bloque 2Kbytes, 10 punteros directos, …), tenemos el archivo *"/home/juan/so/p1/p1.c"* con un tamaño de 3Mbytes e inodo número 6549. Al ejecutar un proceso (con el código indicado a continuación), el usuario efectivo del proceso coincide con el propietario del fichero (*p1.c*), y tiene además los permisos de acceso y lectura a los directorios *raíz*, *home*, *juan*, *so* y *p1*, además del permiso de escritura en el directorio *p1*. Las cachés de datos e inodos están inicialmente vacías y la entrada *so* está en el octavo bloque de su directorio padre (*juan*), mientras las demás entradas están en el primer bloque de sus directorios padre. Contestar lo siguiente referente al siguiente código (se ejecuta desde el directorio */home/juan/so/p1*):

No sistema de ficheiros do problema P1 (tamaño de bloque 2Kbytes, 10 punteiros directos,...), temos o ficheiro *"/home/juan/so/p1/p1.c"* cun tamaño de 3Mbytes e inodo número 6549. Ao executar un proceso (co código que se indica a continuación), o usuario efectivo do proceso coincide co propietario do ficheiro (*p1.c*), e tamén ten permisos de acceso e lectura aos directorios *raíz*, *home*, *juan*, *so* e *p1*, ademais do permiso de escritura no directorio *p1*. Os cachés de datos e inodos están inicialmente baleiros e a entrada *so* está no oitavo bloque do seu directorio pai (*juan*), mentres que as outras entradas están no primeiro bloque dos seus directorios pai. Responder ao seguinte sobre o seguinte código (execútase dende o directorio */home/juan/so/p1*):

In the file system of problem P1 (block size 2Kbytes, 10 direct pointers, ...), we have the file *"/home/juan/so/p1/p1.c"* with a size of 3Mbytes and inode number 6549. When running a process (with the code below), the effective user of the process matches the owner of the file (*p1.c*), and also has access and read permissions to directories *root*, *home*, *juan*, *so* and *p1*, as well as write permission to directory *p1*. The data and inode caches are initially empty and the entry *so* is in the eighth block of its parent directory (*juan*), while the other entries are in the first blocks of their parent directories. Answer the following concerning the following code (it is executed from directory */home/juan/so/p1*):

```
main (){
    struct stat buf, char c;
    int fd1=open("/home/juan/so/p1/p1.c", O_RDONLY);
    chmod("p1.c", 0632);
    lstat ("p1.c", &buf)
    printf("%s", convertir_permisos(buf.st_mode));
                                /* se convierten los permisos a formato rwxrwxrwx y se imprimen*/
                                /* se convirten os permisos a formato rwxrwxrwx e se imprimen*/
                                /* file permissions are converted to format rwxrwxrwx and are printed*/
    link("p1.c", "practica1.c");   /* hard link "practica1.c" */
    symlink("practica1.c", "slink_practica1.c" );   /* link simbólico, symbolic link "slink_practica1.c" */
    int fd2=open("practica1.c", O_RDONLY);
    lseek(fd2, 100000, SEEK_SET);
                        /* SEEK_SET - el desplazamiento se considera a partir del origen del fichero */
                        /* SEEK_SET - o desprazamiento se considera a partir do inicio do ficheiro */
                        /* SEEK_SET - the offset is considered starting from the file origin */
    c=fgetc(fd2);
    close(fd2); close(fd1);
    unlink("p1.c"); }
```

A. ¿Cuál es el número de accesos necesarios a disco, únicamente en el área de datos, en la primera apertura del fichero *p1.c*?: ____ **12**

B. ¿Cuál es el valor asignado al descriptor de fichero *fd2* tras la segunda apertura?: ____ **4**

C. Indica el número de bloques que el S.O. necesita leer en disco para obtener el valor de *c* en *fgetc(fd2*): ____ **2**

D. ¿Cuál es el tamaño del fichero *"slink_practica1.c":*___ ___ (indicar unidad: bytes, Kbytes, Mbytes, …*)* **11 bytes**

E. Indica los permisos del fichero *p1.c* impresos en formato *rwxrwxrwx*: _____ **rw- -wx -w-**

A. Cal é o número de accesos ao disco necesarios, só na área de datos, na primeira apertura do ficheiro *p1.c*?: ____

B. Cal é o valor asignado ao descritor de ficheiro *fd2* despois da segunda apertura?: ____

C. Indica o número de bloques que o S.O. cómpre ler desde o disco para obter o valor de *c* en *fgetc(fd2)*: ____

D. Cal é o tamaño do ficheiro *"slink_practica1.c"*:___ ___ (indicar a unidade: bytes, Kbytes, Mbytes, …)

E. Indica os permisos do ficheiro *p1.c* impresos en formato *rwxrwxrwx*:

A. What is the number of disk accesses required, in the data area only, on the first opening of file *p1.c*? ____

B. What is the value assigned to the file descriptor *fd2* after the second opening? ____

C. Indicate the number of blocks the O.S. needs to read from disk to get the value of *c* in *fgetc(fd2)*: ____

D. What is the size of the file *"slink_practica1.c"*:___ ___ (indicate unit: bytes, Kbytes, Mbytes, ...)

E. Indicate the permissions of file *p1.c* printed in *rwxrwxrwx* format: _____

P4) Indicar si es cierto/falso en cada pregunta.

Cada apartado puntúa 0.1p. Cada respuesta errónea puntúa -0.1p. Cuestiones no respondidas no puntúan. La puntuación mínima de P4 es 0, es decir, en ningún caso P4 lleva a puntuación negativa para el total del examen.

Las 4 primeras preguntas (A, B, C y D) se refieren al código del ejercicio previo (P3).

A. *fgetc* es una llamada al sistema operativo. Cierto/Falso: ____ **F**

B. *unlink* va a eliminar la entrada de directorio *"p1.c"* en su directorio padre pero no libera el inodo 6549. Cierto/Falso: ____ **T**

C. El lincador incorpora el código de *printf* desde la librería estándar de C (versión estática o dinámica de la librería). Cierto/Falso: ____ **T**

D. Al ejecutar la función *printf* de C se incrementa el tiempo de ejecución en modo usuario pero no en modo sistema. Cierto/Falso: ____ **F**

E. Un proceso puede abrir varias veces un fichero, pero debe ser con el mismo modo de apertura. Cierto/Falso: ____ **F**

F. El *Buffer Cache* reduce el número de lecturas físicas, pero no de escrituras físicas, sobre los discos montados. Cierto/Falso: ____ **F**

Indique se é verdadeiro/falso en cada pregunta.

Cada apartado puntúa 0.1p. Cada resposta incorrecta puntúa -0.1p. Cuestións non respondidas non puntúan. A puntuación mínima para P4 é 0, é dicir, en ningún caso P4 ten puntuación negativa para o exame total.

As 4 primeiras preguntas (A, B, C e D) fan referencia ao código do exercicio anterior (P3).

A. *fgetc* é unha chamada ao sistema operativo. Verdadeiro Falso: ____

B. *unlink* eliminará a entrada do directorio *"p1.c"* no seu directorio pai pero non liberará o inodo 6549. Verdadeiro/Falso: ____

C. O enlazador (lincador) incorpora o código de *printf* desde a biblioteca estándar de C (versión estática ou dinámica da biblioteca). Verdadeiro Falso: ____

D. Executar a función C *printf* aumenta o tempo de execución en modo usuario pero non en modo sistema. Verdadeiro Falso: ____

E. Un proceso pode abrir un ficheiro varias veces, pero debe ser co mesmo modo de apertura. Verdadeiro Falso: ____

F. O *Buffer Cache* reduce o número de lecturas físicas, pero non de escrituras físicas, nos discos montados. Verdadeiro Falso: ____

Indicate whether it is true/false for each question.

Each question scores 0.1p. Each wrong answer scores -0.1p. Unanswered questions do not score. The minimum score for P4 is 0, that is, in no case does P4 have a negative score for the total exam.

The first 4 questions (A, B, C and D) refer to the code of the previous exercise (P3).

A. *fgetc* is an operating system call. True/False: ____

B. *unlink* will remove the directory entry *"p1.c"* in its parent directory but does not free inode 6549. True/False: ____

C. The linker incorporates the *printf* code from the C standard library (static or dynamic version of the library). True/False: ____

D. Executing the C *printf* function increases the execution time in user mode but not in system mode. True/False: ____

E. A process can open a file several times, but it must be with the same opening mode. True/False: ____

F. *Buffer Cache* reduces the number of physical reads, but not physical writes, on mounted disks. True/False: ____

Apellidos:_____Nombre:_____ DNI: _____

Examen sin identificación completa NON se califica
Sistemas Operativos - GEI UDC.Xullo 2024. Mem (2.5 puntos)
Esta parte NON admite entrega follas adicionais. Hay que contestar nas páxinas e espacios proporcionados.

1. (1 puntos) Responda Verdadero/Falso ***V/F*** (o no conteste) a cada pregunta (0.05 cada una) sin justificaciones, pero cada respuesta incorrecta anula una correcta. La puntuación mínima es cero para esta pregunta.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
V	V	F	F	F	V	V	V	F	F	F	F	V	F	V	V	V	V	V	V

1. Con Táboa de páxinas invertidas, afórrase memoria para as táboas de páxinas de proceso con respecto aos sistemas de táboas de páxinas.
2. Con Táboa de páxinas invertidas podense xestionar os fallos da páxina do proceso.
3. Cando o lock bit é 0 nunha entrada da táboa de páxinas necesaria para un acceso á memoria, prodúcese un fallo de páxina no acceso a esa páxina.
4. Para un proceso cun espazo virtual de N páxinas, o algoritmo LRU con N marcos sempre (é dicir, en calquera execución do proceso) produce menos erros de páxina que o LRU con N-1 marcos.
5. Para un proceso cun espazo virtual de N páxinas, o algoritmo FIFO con N marcos sempre (é dicir, en calquera execución do proceso) produce menos erros de páxina que FIFO con N-1 marcos.

6. Na pila do proceso están os argumentos da liña de comandos.
7. As variables locais de main() están na pila do proceso.
8. As variables locais das funcións definidas polo usuario están na pila do proceso.
9. O código para as funcións da librería invocadas polo proceso está na pila do proceso.
10. O código para as funcións definidas polo usuario e invocadas está na pila do proceso.

11. As variables estáticas definidas en funcións definidas polo usuario están na pila do proceso.
12. As variables estáticas definidas en main() están na pila do proceso.
13. Para un proceso con varios fíos de execución, cada fío ten a súa propia pila.
14. Considere un sistema de memoria cunha táboa de páxinas dun nivel, se nunha entrada de TP o bit de presenza é 1 (páxina presente) o acceso a esa páxina está garantido
15. Considere un sistema de memoria cunha táboa de páxinas dun nivel, se nunha entrada de TP o bit de presenza é 1 (páxina presente) garántese que non hai fallo de páxina na referencia a esa páxina.

16. Unha chamada de sistema fork en sistemas Unix antigos sen un mecanismo de copy-on-write, copia a táboa de páxinas do proceso pai no proceso fillo.
17. Unha chamada de sistema fork nos sistemas Unix modernos cun mecanismo de copy-on-write, copia a táboa de páxinas do proceso pai no proceso fillo.
18. Nun sistema con paxinación por demanda pura, sempre se produce un fallo de páxina cando se executa a primeira instrución.
19. Unha táboa de páxinas de varios niveis normalmente reduce a cantidade de memoria necesaria para almacenar táboas de páxinas en comparación cunha táboa de páxinas dun só nivel.
20. Se o número de marco vén dado por 8 bits e as páxinas son de 4Kbytes, os enderezos físicos son de 20 bits.

2. **tres apartados a 0.70, b 0.40, c 0.40**

a) (0.70 puntos) Para un sistema de tabla de páginas multinivel con un direccionamiento de memoria física de 8GBytes, tamaño de página de 4Kbytes y tamaño de una entrada de tabla de páginas de 8 bytes. ¿Cuántos niveles son necesarios para la tabla de páginas si el sistema tiene direcciones virtuales de 30 bits, si tanto la página del nivel raíz como las páginas de los otros niveles ocupan todo su espacio con entradas de la tabla de página? Debe indicar los cálculos para que la respuesta puntúe.

Número de niveles:_2__

Para un sistema de táboas de páxinas multinivel cun enderezo de memoria física de 8 GB, un tamaño de páxina de 4 Kbytes e un tamaño dunha entrada de táboa de páxinas de 8 bytes. Cantos niveis son necesarios para a táboa de páxinas se o sistema ten enderezos virtuais de 30 bits, se tanto a páxina de nivel raíz como as páxinas dos outros niveis ocupan todo o seu espazo con entradas da táboa de páxinas? Debes indicar os cálculos para que a resposta conte.

Número de niveis:_2__

For a multilevel page table system with a physical memory address of 8GBytes, page size of 4Kbytes, and size of a page table entry of 8 bytes. How many levels are necessary for the page table if the system has 30-bit virtual addresses, if both the root level page and the pages at the other levels take up all their space with page table entries? You must indicate the calculations for the answer to count.

Number of levels:___2____

páginas de 4Kbytes= 2^{12} bytes

2^{12} bytes / 2^{3} bytes = 2^{9} entradas por tabla de páginas

un TP de un nivel puede direccionar 2^{9} entradas x 2^{12} bytes = 2^{21} bytes

con dos niveles de TP se pueden direccionar:

2^{9} entradas x 2^{9} entradas x 2^{12} bytes = 2^{30} bytes

por tanto son suficientes dos niveles para gestionar direcciones virtuales de 30 bits

b) (0.40 puntos) Para la solución anterior, ¿de cuántos bits de control (bit referencia, bit R/W, dirty bit, etc) y bits no usados (se pide la suma total de ambos tipos), se dispone en una entrada de tabla de página de cada nivel? Debe indicar los cálculos para que la respuesta puntúe.

Para a solución anterior, cantos bits de control (bit de referencia, bit R/W, dirty bit, etc.) e bits non utilizados (se solicita a suma total de ambos os tipos) están dispoñibles nunha entrada da táboa de páxinas de cada nivel? Debes indicar os cálculos para que a resposta conte.

For the above solution, how many control bits (reference bit, R/W bit, dirty bit, etc.) and unused bits (the total sum of both types is requested) are available in a page table entry of each level? You must indicate the calculations for the answer to count.

páginas de 4Kbytes= 2^{12} bytes

memoria física de 8Gbytes = 2^{33} bytes

2^{33} bytes / 2^{12} bytes = 2^{21} páginas físicas, se necesitan 21 bits en la entrada de la TP para poder direccionar las 2^{21} páginas físicas

entradas en la TP de 8 bytes = 64 bits, quedan 64-21= 43 bits no usados en cada entrada de la TP, lo mismo para todos los niveles.

c) (0.40 puntos) Imagine que con la solución anterior el procesador dispone de caché de datos e instrucciones y TLB, en este caso, para una operación de lectura de un byte de memoria (debe dar la explicación correcta para que la respuesta puntúe),

¿cuál sería el número mínimo de accesos a memoria? : ___0____

¿cuál sería el número máximo de accesos a memoria?:____3____

Imaxina que coa solución anterior o procesador ten unha caché de datos e instrucións e TLB, neste caso, para unha operación de lectura dun byte de memoria (debes dar a explicación correcta para que a resposta se puntue),

Cal sería o número mínimo de accesos á memoria? :____0___

Cal sería o número máximo de accesos á memoria?:____3____

Imagine that with the previous solution the processor has a data and instruction cache and TLB, in this case, for a read operation of a byte of memory (you must give the correct explanation for the answer to score),

What would be the minimum number of memory accesses? :___0____

What would be the maximum number of memory accesses?:____3____

número mínimo: el mapping página lógica a página física se encuentra en la TLB, y el byte de memoria en la caché, 0 accesos a memoria

número máximo: 2 accesos a memoria para cada nivel de la TP, 1 acceso a memoria para conseguir el byte en memoria, total 3 accesos a memoria

APELLIDOS(Mayúscula**):**_____**NOMBRE:**_____**DNI.** _____

Sistemas Operativos (PROCESOS+E/S) – Grado Ingeniería Informática UDC. Julio 2024.

-Examen sin identificación completa NO SE CALIFICA.

-Apellidos en MAYUSCULAS (0.25)

-Esta parte NO admite entrega de hojas adicionales. Hay que contestar en los espacios proporcionados.

Q1.-(1 punto) Responda Verdadero/Falso (o no conteste) a cada pregunta (correcta 0.07; incorrecta -0.07). La puntuación mínima es cero para esta pregunta. (Rellenar en el espacio correspondiente de la siguiente tabla).

Answer V/F (True/False) or leave the answer blank (correct answer: 0.07; wrong answer: -0.07). Minimum score for this question: 0. Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder (resposta correcta: +0.07; resposta incorrecta: -0.07). Puntuación mínima para esta pregunta: 0

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
V	**F**	**V**	**V**	**V**	**V**	**F**	**F**	**F**	**V**	**V**	**F**	**F**	**F**	**F**

1.-La cuatro capas del software de e/s son: software de nivel de usuario, software independiente dismositivo, manejador dispositivo y manejador de interrupciones.
2.-Un dispositivo "polling" genera interrupciones
3.-La e/s mediante DMA puede usuarse con dispositivos mapeados en memoria.
4.-En un sistema que tiene espacio e/s separado puede haber también dispositivos mapeados en memoria.
5.-Compartir memoria con *shmget* puede hacerse entre procesos con distinto *uid*
6.- En le caso de que el ejecutable tenga el bit setuid, exec() cambia la credencial efectiva y la salvada
7.-Todo proceso es una sucesion de rafagas de CPU y e/s. En algunos S.O. se permite que un proceso termine con una ráfaga de e/s.
8.-Una planificación round-robin puede producir inanición de procesos con una ráfaga mayor que el quanto
9.-La prioridad es un entero y su rango de valores depende, en parte, del microprocesador, p.e. una versión de linux para procesadores intel probablemente tenga distintos valores para las prioridades que la misma versión para procesadores superSPARC
10.-En un sistema tipo UNIX, la Tabla de Ficheros Abiertos del sistema es parte los datos de kernel
11.-Que cada proceso tenga su propia pila del kernel es condición necesaria para que el kernel del S.O. sea reentrante
12.-En un sistema multiprocesador, el kernel no puede ser reentrante
13.-Todos los procesos de un sistema tipo UNIX, al comenzar, tienen el mismo conjunto de variables de entorno
14.-Para implementar un S.O. tipo UNIX para un determinado microprocesador, es condición necesaria que el procesador tenga la instrucción "terminar proceso" o equivalente
15.-Al crear un proceso mediante exec, exec devuelve el pid del proceso creado al proceso padre

1-The four layers of I/O software are: user-level software, device independent, device driver and interrupt handler.
2.-A polling device does generate interruptions
3.-DMA I/O can be used with memory mapped devices.
4.-In a system that has separate I/O space can also have memory-mapped devices.
5.-Sharing memory with shmget can be done between processes with different uid
6.- If the executable has the setuid bit, exec() changes the effective and saved credentials
7.-Every process is a succession of CPU and I/O bursts. In some OSs A process is allowed to terminate with a burst of I/O.
8.-A round-robin schedule can produce process starvation for a process with a burst greater than the quantum
9.-The priority is an integer and its range of values depends, in part, on the microprocessor, e.g. A version of Linux for Intel processors will probably have different priority values than the same version for SuperSPARC processors.
10.-In a UNIX-type system, the system's Open File Table is part of the kernel data
11.-That each process has its own kernel stack is a necessary condition for the OS kernel be reentrant
12.-In a multiprocessor system, the kernel cannot be reentrant
13.-All processes of a UNIX-type system, upon starting, have the same set of environment variables
14.-To implement an UNIX type o.s. for a given microprocessor, it is a necessary condition that the processor has the "end process" instruction or equivalent
15.-When creating a process using exec, exec returns the pid of the created process to the parent process

1-As catro capas do software de E/S son: software a nivel de usuario, soft independente de dispositivo, manexador de dispositivo e manexador interrupción.
2.-Un dispositivo "polling" xera interrupcións
3.-DMA I/O pódese usar con dispositivos mapeados con memoria.
4.-Nun sistema que teña espazo de E/S separado pode haber tamén dispositivos mapeados en memoria.
5.-Compartir memoria con shmget pódese facer entre procesos con distinto uid
6.- Se o executable ten o bit setuid, exec() cambia as credencial efectiva e gardada
7.-Todo proceso é unha sucesión de ráfagas de CPU e E/S. Nalgúns S.O. permítese que un proceso remate cunha ráfaga de E/S.
8.-Unha planificación round-robin pode producir inanición de procesos cunha ráfaga maior que o cuanto
9.-A prioridade é un número enteiro e o seu rango de valores depende, en parte, do microprocesador, p.e. unha versión de Linux para procesadores Intel probablemente terá valores de prioridade diferentes que a mesma versión para procesadores SuperSPARC.
10.-Nun sistema tipo UNIX, a táboa de ficheiros abertos do sistema é parte dos datos do núcleo
11.-Que cada proceso teña a súa propia pila de núcleo é unha condición necesaria para que o núcleo do SO. sexa reentrante
12.-Nun sistema multiprocesador, o núcleo non pode ser reentrante
13.-Todos os procesos dun sistema tipo UNIX, ao iniciarse, teñen o mesmo conxunto de variables de ambiente
14.-Implementar un SO tipo UNIX para un microprocesador dado, é unha condición necesaria que o procesador teña a instrución "terminar proceso" ou equivalente
15.-Ao crear un proceso usando exec, exec devolve o pid do proceso creado ao proceso pai

APELLIDOS(Mayúscula**):**_____**NOMBRE:**_____**DNI.** _____

Q2.-(0.75 puntos) La salida de ls es la que se muestra en el cuadro, donde denominaremos mensaje1 a "**/home: drwx------ 95 usuario usuario 4096 Jun 21 10:23 usuario**" y mensaje2 a "**ls: cannot open directory '/root': Permission denied**" es decir, mensaje1 es lo que va a la salida estandar y mensaje2 lo que va al error estandar. Se supone que ninguna de las llamadas al sistema producen ningun error y que los códigos tienen los includes correspondientes.

```
usuario@unixmachine:~$ ls -l /root /home
/home:
drwx------ 95 usuario  usuario   4096 Jun 21 10:23 usuario
ls: cannot open directory '/root': Permission denied
```

Código 1

```
main()
{
   char *arg[]={"ls","-l","/root","/home",NULL};
   execv("/bin/ls", arg);
}
```

Código 2

```
main()
{  int df1; df2;
   char *arg[]={"ls","-l","/root","/home",NULL};
   if (fork()==0) {
      df1=open("out.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
      df2=open("err.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
      close(1); dup (df2);
      close(2); dup(1);
   }
   execv("/bin/ls", arg);

}
```

Código 3

```
main()
{  int df1; df2;
   char *arg[]={"ls","-l","/root","/home",NULL};

   df1=open("out.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
   df2=open("err.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
   close(1); dup (df2);
   close(2); dup(1);
   execv("/bin/ls", arg);
}
```

Código 4

```
main()
{  int df1; df2;
   char *arg[]={"ls","-l","/root","/home",NULL};

   df1=open("out.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
   df2=open("err.txt",O_WRONLY,O_CREAT, O_TRUNC,0777);
   close(1); dup (df2);
   close(2); dup(1);
   if (fork()==0)
      execv("/bin/ls", arg);

}
```

Rellenar el siguiente cuadro indicando lo que va a los ficheros "out.txt", "err.txt" y a la pantalla, tras la ejecución de cada uno de los códigos., tal como aparece para el código1

	fichero "out.txt"	fichero "err.txt"	pantalla
Código 1	------	------	**mensaje1** **mensaje2**
Código 2	------	**mensaje1** **mensaje2**	**mensaje1** **mensaje2**
Código 3		**mensaje1** **mensaje2**	
Código 4		**mensaje1** **mensaje2**	

EXPLICACION: El código de redirección redirecciona salida y error estándar al fichero err.txt.
Código 2: La redirección se hace en el hijo, el padre ejecuta el exec sin redirección y el hijo con redirección
Código 3: Se hace la redirección y luego el exec
Código 4: El padre hace la redirección antes de crear el hijo. El hijo hereda la redirección y hace el exec

APELLIDOS(Mayúscula**):**_____**NOMBRE:**_____**DNI.** _____

Q3.-(0.5 puntos) Se muestra una planificaciónen un sistema monoprocesador. A CPU scheduling is shown on a uniprocessor system. Móstrase unha programación nun sistema uniprocesador.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A	B	C	D	A	E	E	E	E	E	E	C	C	D	B	A	A	A	A	A

¿Es posible que se trate de una planificación round robing de quanto 2? Is it possible that this is a quantum 2 round robin planning? É posible que se trate dunha planificación round robin quantum 2? Elegir a) (**Si**) o b) (**No**) y contestar lo que se pide en ese caso. Choose a)(**Yes**) or b) (**No**) and answer accordingly. Escolle a) (**Si**) ou b) (**Non**) e responde ao que se pregunta nese caso.

a) Si. Indique una posible duracrión de las ráfagas de CPU y e/s de los procesos asi como sus instantes de llegada.
Yes Indicate a possible duration of the CPU and I/O bursts of the processes as well as their arrival times.
Si. Indique unha posible duración das ráfagas de CPU e E/S dos procesos así como os seus tempos de chegada.

b) No. Justifíquese por qué. **No,** please explain why. **Non,** Xustifica por qué

SI, es posible. La solución mas simple (no la única) seria:

PROCESO	**LLEGADA**	**RAFAGAS**
A	**0**	**1-(3)-1-(10)-5**
B	**1**	**1-(12)-1**
C	**2**	**1-(8)-2**
D	**3**	**1-(9)-1**
E	**5**	**6**

Los números entre paréntesis representan ráfagas de e/s y los sin paréntesis ráfagas de CPU, p.e 1-(2)-3 representa una ráfaga de CPU de 1, seguida de una ráfaga de e/s de 2, seguida de una ráfaga de CPU de 3

La solución no es única, por ejemplo

***ABCD llegan todos en el instante 0 en el orden ABCD y E en el instante 5, (resto de datos igual)**

***ABCD llegan todos en el instante 0 en el orden ABCD, la primera ráfaga de e/s de A es de 1, y E llega en el instante 3 (resto de datos igual)**

***A llega en el instante 0, BCD llegan en el instante 1 en el orden BCD, la primera ráfaga de e/s de A es de 1 y E llega en el instante 3 (resto de datos igual)**

***.....**

APELLIDOS(Mayúscula**):**_____**NOMBRE:**_____**DNI.** _____

Q4.-(0.5 puntos) Sea el siguiente código en C, con todos los includes necesarios, que compila correctamente y que produce un ejecutable a.out.
Consider the following C code, with all the necessary includes, that compiles correctly and the corresponding a.out executable file.
Sexa o seguinte código en C, con todas os includes necesarios, que compila correctamente e que produce un ficheiro executable a.out.

Tanto a.out como f1.txt son del usuario u2, a.out es ejecutado por un usuario u1, desde el mismo directorio donde están a.out y f1.txt. Completar el siguiente cuadro indicando las credenciales reales y efectivas del proceso que ejecuta a.out y si alguno de los descriptores df1 o df2 es -1 dependiendo de los permisos de a.out y f1.txt.
Both a.out and f1.txt are from user u2, a.out is executed by user u1, from the same directory where a.out and f1.txt are. Complete the following table indicating the real and effective user credentials of the process running a.out and if any of the df1 or df2 descriptors is -1 depending on the permissions of a.out and f1.txt.
Tanto a.out como f1.txt son do usuario u2, a.out é executado polo usuario u1, dende o mesmo directorio onde están a.out e f1.txt. Completa a seguinte táboa indicando as credenciais de usuario reais e efectivas do proceso que executa a.out e se algún dos descritores df1 ou df2 é -1 dependendo dos permisos de a.out e f1.txt

```
int main (int argc, char *argv[])
{

    int df1, df2;
    df1=open ("./f1.txt", O_RDONLY);
    df2=open ("./f1.txt", O_RDWR);
    printf ("%d, %d\n", df1, df2);
}
```

a.out	f1.txt	ruid	euid	df1=-1?	df2=-1?
rwxrwxrwx	rwxrwxrwx	u1	u1	no	no
rwsr-xr-x	rwxr-xr-x	u1	u2	no	no
rwsr-xr-x	rwxr--r--	u1	u2	no	no
rwsr-xr-x	rwsr-xr-x	u1	u2	no	no
rwsr-xr-x	rwsr--r--	u1	u2	no	no
rwsr-xr-x	rws------	u1	u2	no	no
rwxr-xr-x	rwxr-xr-x	u1	u1	no	si
rwxr-xr-x	rwxr--r--	u1	u1	no	si
rwxr-xr-x	rwsr-xr-x	u1	u1	no	si
rwxr-xr-x	rwsr--r--	u1	u1	no	si
rwxr-xr-x	rws------	u1	u1	si	si

EXPLICACION: La credencial real es SIEMPRE la de quien ejecuta el fichero. Si el ejecutable tiene el bit setuid, exec cambia la credencial efectiva a la del propietario del ejecutable (o sea, cuando los permisos del ejecutable son rws-r-xr-x, la credencial efectiva es u2).

Hay dos aperturas, df1 en modo solo lectura, y df2 en modo lectura/escritura. Cuando la credencial efectiva es u2 se miran los permisos de propietario y cuando la credencial efectiva es u1 se miran los permisos de resto ya que no sabemos si u1 pertenece al grupo del fichero (no importaría los permisos de resto y de grupo son los mismos)