Apellidos:_____ Nombre:_____ DNI: _____

Sistemas Operativos – Grado Ingeniería Informática UDC. Enero 2023
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: 0.5, P3: 0.3, P4:0.6

P1)

Un sistema de archivos tipo UNIX tiene un tamaño de bloque de 4Kbytes, i-nodos con 10 direcciones directas, una indirecta simple, una indirecta doble y una indirecta triple. Utiliza direcciones de bloque de 8 bytes. Calcular cuántos bloques de disco son necesarios (en el área de datos) para representar un archivo de tamaño 1 Gbytes + 6 Mbytes + 10 Kbytes. Discriminar cuántos bloques son de datos y cuántos de índices

Un sistema de ficheiros tipo UNIX ten un tamaño de bloque de 4Kbytes, i-nodos con 10 enderezos directos, un indirecto simple, un indirecto dobre e un indirecto triple. Usa enderezos de bloque de 8 bytes. Calcular cantos bloques de disco son necesarios (na área de datos) para representar un ficheiro de tamaño 1 Gbytes + 6 Mbytes + 10 Kbytes. Discriminar cantos bloques son de datos e cantos de índices.

A UNIX file system has a block size of 4Kbytes, i-nodes with 10 direct addresses, a single indirect, a double indirect and a triple indirect address. It uses 8-byte block addresses. Calculate how many blocks are needed (in the data area) to represent a file with size 1 Gbytes + 6 Mbytes + 10 Kbytes. Discriminate how many blocks are data blocks and how many index blocks.

Nº bloques de datos/Nº bloques de datos/Number of data blocks: 0.25 Gbloques + 1.5 Kbloques + 3 bloques (0.30p)

Nº de bloques de índices/Nº de bloques de índices/Number of index blocks: 518 (0.25p)

Fragmentación interna del fichero/Fragmentación interna do ficheiro/ Internal file fragmentation (0.05p): 2048 (bytes)

P2) Un proceso abre 2 veces el archivo *"/home/user1/so/practicas/p1.c"*, cuyo número de *hard links* inicial es 2. El tamaño del fichero es de 1Gbytes + 6Mbytes + 10Kbytes y se suponen los datos referentes al sistema de ficheros del ejercicio P1 (tamaño bloque, punteros acceso directos e indirectos). El usuario efectivo del proceso coincide con el propietario del fichero, y tiene además los permisos de acceso a los directorios *raíz*, *home*, *user1, so* y *practicas*. Las cachés de datos e inodos están inicialmente vacías. Indicar lo siguiente referente al trozo de código:

Un proceso abre duas veces o ficheiro *"/home/user1/so/practicas/p1.c"*, cuxo número de *hard links* inicial é 2. O tamaño do ficheiro é de 1Gbytes + 6Mbytes + 10Kbytes e se supoñen os datos referentes ao sistema de ficheiros do exercicio P1 (tamaño bloque, punteiros acceso directos e indirectos). O usuario efectivo do proceso coincide co propietario do ficheiro e tamén ten os permisos de acceso os directorios *raíz, home*, *user1, so* e *practicas*. Os cachés de datos e inodos están inicialmente baleiros. Indique o seguinte sobre o anaco de código:

A process opens twice the file "*/home/user1/so/practicas/p1.c*", whose initial number of *hard links* is 2. The file size is 1Gbytes + 6Mbytes + 10Kbytes and the data referring to the file system from exercise P1 is assumed (block size, direct and indirect access addresses). The effective user of the process coincides with the owner of the file, and also has the access permissions to the directories *root*, *home*, *user1*, *so* and *practicas*. The data and inode caches are initially empty. Indicate the following regarding the piece of code:

```
chmod("/home/user1/so/practicas /p1.c", 0654);
link("/home/user1/so/practicas /p1.c", "/home/user1/so/practicas /p1_hlink.c");
symlink("/home/user1/so/practicas/p1_hlink.c", "/home/user1/so/practicas/p1_slink");
                                                    /*crea link simbólico p1_slink */
                                                    /* crea link simbólico p1_slink */
                                                    /* symbolic link p1_slink is created*/
int fd1=open("/home/user1/so/practicas /p1.c", O_RDONLY); /* es la primera apertura de fichero del proceso */
                                                    /* é a primeira apertura de ficheiro do proceso */
                                                    /* it is the first file open of the process*/
int fd2=open("/home/user1/so/practicas /p1_hlink.c", O_RDWR);
lseek(fd1, 1.000.000, SEEK_SET);/*SEEK_SET indica que el desplazamiento se considera a partir del origen del fichero */
                                /* SEEK_SET indica que o desprazamento considérase desde a orixe do ficheiro */
                                /* SEEK_SET specifies that the offset is considered from the origin of the file */
char c1=fgetc(fd1);
lseek(fd2, 8.000, SEEK_SET);
char c2=fgetc(fd2);
close(fd1); close(fd2);
unlink("/home/user1/so/practicas /p1_slink");
```

Cada apartado puntúa 0.1p/ Each question scores 0.1p.

A. ¿Cuál es tamaño del fichero *p1_slink*?: 35 bytes (Indicar bytes/Mbytes/Gbytes)
¿Cal é o tamaño do ficheiro *p1_slink*?: (Indicar bytes/Mbytes/Gbytes)
What is the size of file *p1_slink*?: _____ (Specify bytes/Mbytes/Gbytes)

B. Indica el número de bloques que el SO necesita leer en disco para obtener el valor de *c1* en *fgetc(fd1)*:
Indica o número de bloques que o SO cómpre ler desde o disco para obter o valor de *c1* en *fgetc (fd1)*:
Indicate the number of blocks that the OS requires to read from disk to get the value of *c1* in *fgetc (fd1): 2*

C. Indica el número de bloques que el SO necesita leer en disco para obtener el valor de *c2* en *fgetc(fd2)*:
Indica o número de bloques que o SO cómpre ler desde o disco para obter o valor de *c2* en *fgetc (fd2)*:
Indicate the number of blocks that the OS requires to read from disk to get the value of *c2* in *fgetc (fd2): 1*

D. Indica los permisos del fichero *p1.c* (después de ejecutar *chmod*) en formato *rwxrwxrwx*: rw- r-x r--
Indica os permisos do ficheiro *p1.c* (despois de executar *chmod*) en formato *rwxrwxrwx*:
Indicate the file permissions of *p1.c* (after running *chmod*) in *rwxrwxrwx* format:

E. ¿Cuál es el número de *hard links* de *p1.c* después de ejecutar *unlink*?: 3
¿Cal é o número de ligazóns duros (*hard links*) de *p1.c* despois de executar *unlink*?:
What is the number of *hard links* of *p1.c* after running *unlink*?: _____

P3) En el sistema de archivos UNIX previo (tamaño de bloque de 4 Kbytes), el *superbloque* ocupa 7 bloques de su partición de disco. Los bloques se numeran a partir del bloque 0. Un fichero “datos” tiene asociado el *inodo* 642 y este inodo está en el bloque (lógico) 21 desde el principio de la partición. El tamaño del *inodo* es de 64 bytes. Calcula el tamaño (en Kbytes, no bloques) del *boot*.

Tamaño del *boot*: 16 (Kbytes)

No sistema de ficheiros UNIX previo (tamaño de bloque de 4 Kbytes), o *superbloque* ocupa 7 bloques da súa partición de disco. Os bloques númeranse a partir do bloque 0. Un ficheiro “datos” ten asociado o inodo 642 e este inodo está no bloque (lóxico) 21 desde o inicio da partición. O tamaño do inodo é de 64 bytes. Calcula o tamaño (en Kbytes, non bloques) do *boot*.

Tamaño do *boot*: _____ (Kbytes)

In the previous UNIX file system (4Kbytes block size), the superblock occupies 7 blocks of its disk partition. Blocks are numbered starting from block 0. A file “datos” has the inode 642 associated and this inode is in the (logical) block 21 from the beginning of the partition. The inode size is 64 bytes. Calculate the size (in Kbytes, not blocks) of the *boot*.

Size of the *boot*: _____ (Kbytes)

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

Cada apartado puntúa 0.1p. Cada respuesta errónea puntúa -0.1. 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.

A. Cuando un proceso ejecuta en modo usuario, se puede ejecutar todo el repertorio de instrucciones máquina del procesador. **F**

B. Al ejecutar la función de entrada/salida de C *perror()*, el proceso está ejecutando exclusivamente en modo usuario. **F**

C. El código binario de *chmod()* se encuentra en la librería estándar de C (*libC*). **F**

D. Las librerías dinámicas se usan para tener ejecutables con código autocontenido. **F**

E. Con un sistema de ficheros con asignación indexada de bloques (como la de Unix), puede existir fragmentación externa. **F**

F. El Buffer Cache permite reducir operaciones de lectura sobre los discos montados, pero no operaciones de escritura en los mismos. **F**

Indique se é verdadeiro/falso en cada pregunta.

Cada apartado puntúa 0.1p. Cada resposta incorrecta puntúa -0.1. 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.

A. Cando un proceso executa en modo usuario, pódese executar todo o conxunto de instrucións máquina do procesador. ____

B. Ao executar a función de entrada/saída de C *perror()*, o proceso execútase exclusivamente en modo usuario. ____

C. O código binario de *chmod()* se atopa na librería estándar de C (*libC*). ____

D. As librerías dinámicas utilízanse para ter executables con código auto-contido. ____

E. Cun sistema de ficheiros con asignación de bloques indexada (como a de Unix), pode haber fragmentación externa. ____

F. O Buffer Cache permite reducir as operacións de lectura nos discos montados, pero non operacións de escritura nos mesmos. ____

Indicate whether it is true/false for each question.

Each question scores 0.1p. Each wrong answer scores -0.1. 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.

A. When a process runs in user mode, the processor's entire set of machine instructions can be run. ____

B. When running the C input/output function *perror()*, the process is running exclusively in user mode. ____

C. The binary code for *chmod()* is in the standard C library (*libC*). ____

D. Dynamic libraries are used to have executables with self-contained code. ____

E. With a file system with indexed block allocation (like Unix's), there may be external fragmentation. ____

F. The Buffer Cache allows to reduce read operations on mounted disks, but not write operations on them. ____

Apellidos:_____Nombre:_____ DNI: _____

Examen sin identificación completa NON se califica
Sistemas Operativos - GEI UDC. Xaneiro 2023. Memoria (2 puntos)
Esta parte NON admite entrega follas adicionais. Hay que contestar nas páxinas e espacios proporcionados.

1. (0.5 puntos) **Cada una de las preguntas siguientes son de responder V/F (Verdadero/Falso) o no responder. Respuesta correcta: +0.1; Respuesta incorrecta: -0.1. (0.1 x 10 = 1 punto) Puntuación mínima para esta pregunta: 0.**
Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder. Resposta correcta: +0.1; Resposta incorrecta: -0.1. (0.1 x 10 = 1 punto) Puntuación mínima para esta pregunta: 0.
Answer T/F (True/False) or leave the answer blank. Correct answer: 0.1; Wrong answer: -0.1 (0.1 x 10 = 1 point) Minimum score for this question: 0.

1	2	3	4	5
V	V	V	V	V

1. **Un sistema con segmentación pero sin paginación, puede tener memoria virtual.**
Un sistema con segmentación pero sen paxinación pode ter memoria virtual.
A system with segmentation but no paging may have virtual memory..

2. A la vuelta de una llamada al sistema fork(), el proceso padre y el proceso hijo tienen su propio espacio de direcciones virtuales.
A volta dunha chamada ao sistema fork(), o proceso pai e o proceso fillo teñen cada un o seu propio espazo de enderezos virtuais
Upon return of a fork() system call, the parent process and the child process each have their own virtual address space.

3. Un servicio de fallo de página de un proceso necesita poner al proceso en el estado de bloqueado mientras se sirve el fallo de página..
Un servizo de fallo de páxina de un proceso debe poñer o proceso no estado bloqueado mentras se sirve o fallo de páxina.
A process page fault service needs to put the process in the blocked state while the page fault is being served.

4. Un sistema operativo que implementa la política del Working Set para la gestión de la memoria virtual, puede llegar a suspender procesos y pasarlos al espacio de intercambio.
Un sistema operativo que implementa a política de Working Set para xestionar a memoria virtual pode suspender procesos e pasalos ao espazo de intercambio.
An operating system that implements the Working Set policy for managing virtual memory can suspend processes and pass them to the swapping space.

5. Una ventaja de un sistema de tablas de páginas multinivel con respecto uno de un nivel es el menor espacio en memoria para las tablas de páginas de los procesos.
Unha vantaxe dun sistema de táboas de páxinas de varios niveis fronte a un dun nivel é o menor espazo de memoria para as táboas de páxinas dos procesos.
One advantage of a multi-level page table system over a one-level one is less memory space for process page tables.

2. (0.75 puntos) Para cada una de las preguntas respuesta correcta: +0.05.
El código que se muestra se compila y ejecuta correctamente. Debe indicar para cada caso lo que se muestra por pantalla eligiendo una de las respuestas posibles de entre las siguientes:
1) una dirección del Espacio Direcciones Virtuales del proceso, area de stack
2) una dirección del Espacio Direcciones Virtuales del proceso, area de datos dinámicos
3) una dirección del Espacio Direcciones Virtuales del proceso, areas de datos globales y estáticos
4) una dirección del Espacio Direcciones Virtuales del proceso, areas de código propio o librerías
5) una dirección del Espacio Direcciones Virtuales del kernel, area de stack
6) una dirección del Espacio Direcciones Virtuales del kernel, area de datos dinámicos
7) una dirección del Espacio Direcciones Virtuales del kernel, areas de datos globales y estáticos
8) una dirección del Espacio Direcciones Virtuales del kernel, areas de código propio o librerías
9) una dirección física
10) un valor que no es una dirección válida del proceso ni del kernel (ni lógica, ni física)
Para cada unha das preguntas: Resposta correcta: +0.05.
O código que se mostra compila e execútase correctamente. Debes indicar para cada caso o que se mostra na pantalla escollendo unha das posibles respostas entre as que siguen:
1) un enderezo do Espazo de Enderezos Virtual do proceso, área de pila
2) un enderezo do Espazo de Enderezos virtual do proceso, área de datos dinámicos
3) un enderezo do Espazo de Enderezos virtuais do proceso, áreas de datos globais e estáticos
4) un enderezo do Espazo de Dirección Virtual do proceso, áreas de código propio ou bibliotecas
5) un enderezo do Espazo de Enderezos Virtual do núcleo, área de pila
6) un enderezo do Espazo de Enderezos Virtual do núcleo, área de datos dinámicos
7) un enderezo do Espazo de Enderezos Virtual do núcleo, áreas de datos globais e estáticos
8) un enderezo do Espazo de Enderezos Virtual do núcleo, áreas de códigos propios ou bibliotecas
9) un enderezo físico
10) un valor que non é un enderezo válido do proceso nin do núcleo (nin lóxico nin físico)
For each of the following questions, correct answer: +0.05
The code shown compiles and runs successfully. You must indicate for each case what is shown on the screen by choosing one of the possible answers from among those shown:
1) an address from the Virtual Address Space of the process, stack area
2) an address from the Virtual Address Space of the process, dynamic data area
3) an address Space Virtual Addresses of the process, global and static data areas
4) an address of the Virtual Address Space of the process, own code areas or libraries
5) an address from the Virtual Address Space of the kernel, stack area
6) an address from the Virtual Address Space of the kernel, dynamic data area
7) an address from the Virtual Address Space of the kernel, global and static data areas
8) an address from the Virtual Address Space of the kernel, own code areas or libraries
9) a physical address
10) a value that is not a valid process or kernel address (neither logical nor physical)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	1	1	1	1	3	3	3	3	1	1	2	2	4	4

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>

double t[3] = {1.0, 2.0, 3.0};

void f1()
{
   static double sd;
   char *s1 = malloc(4096);
   char *s2;

   sd = t[1];

   printf ("I: &sd        %p\n",&sd);


   strcpy(s1, "alfa");
   s2 = strdup("beta");

   printf ("J: &s1        %p\n",&s1);
   printf ("K: &s2         %p\n",&s2);

   printf ("L: s1         %p\n",s1);
   printf ("M: s2         %p\n",s2);


   printf ("N:  strcpy    %p\n",strcpy);
   printf ("O:  exit      %p\n",exit);
}

int main(int argc, char *argv[])
{
  int args;
  int *p;

  args=argc;
  p=&argc;

  printf ("A:  p         %p\n", p);
  printf ("B:  &p        %p\n",&p);
  printf ("C:  &args     %p\n",&args);

  printf ("D  argv       %p\n", argv);
  printf ("E  &argv[0]   %p\n", &argv[0]);


  printf ("F:  t         %p\n", t);
  printf ("G:  &t[0]     %p\n", &t[0]);
  printf ("H:  &t        %p\n", &t);


  f1();
  exit(0);
}
```

3. (0.75 puntos: 0.15 por apartado) Considere una sistema de paginación en dos niveles. Las páginas son de 4 Kbytes. El nivel 0 tiene una página raíz y el nivel 1 tiene las páginas necesarias para el espacio de direcciones virtuales de 32 bits. Cada página de la tabla de páginas de ambos niveles contiene 1024 entradas, 1 byte de de cada entrada está reservado para permisos y otros usos del SO.
Considere un sistema de paxinación de dous niveis. As páxinas son de 4 Kbytes. O nivel 0 ten unha páxina raíz e o nivel 1 ten as páxinas necesarias para o espazo de enderezos virtuais de 32 bits. Cada páxina da táboa de páxinas de ambos niveis contén 1024 entradas, 1 byte de cada entrada está reservado para permisos e outros usos do sistema operativo.
Consider a two-tier paging system. The pages are 4 Kbytes. Level 0 has a root page, and level 1 has the pages required for the 32-bit virtual address space. Each page of the page table of both levels contains 1024 entries, 1 byte of each entry is reserved for permissions and other OS uses.

El resultado final y los cálculos o explicacioes deben ser correctos y suficientes para puntuar cada pregunta.
O resultado final e os cálculos ou explicacións deberán ser correctos e suficientes para puntuar cada pregunta.
The final result and the calculations or explanations must be correct and sufficient to score each question

a) **Indique el formato de las direcciones virtuales.**
Indica o formato dos enderezos virtuais.
Show the format of virtual addresses.

10 bits entrada TP nivel 0, 10 bits entrada TP nivel 1, 12 bits offset páxina

b) **¿Cúal es el tamaño del espacio de direcciones virtuales?**
Cal é o tamaño do espazo de enderezos virtuais?
What is the size of the virtual address space?

2^{32} bytes = 4 Gigabytes

c) **¿Cúal es el tamaño máximo posible del espacio de direcciones físicas con estas especificaciones?**
Cal é o tamaño máximo posible do espazo de enderezos físicos con estas especificacións?
What is the maximum possible size of the physical address space with these specifications?

2^{12} bytes/páxina / 2^{10} entradas = 4 bytes por entrada, 1 byte reservado, 3 bytes para direccionar frames
2^{24} frames de 2^{12} bytes = 64 Gigabytes

d) **¿Cual sería el espacio ocupado por la tabla de páginas para un proceso que use todo su espacio de direcciones virtuales?**
Cal sería o espazo que ocupa a táboa de páxinas para un proceso que utiliza todo o seu espazo de enderezos virtuais?
What would be the space occupied by the page table for a process that uses all of its virtual address space?

1 páxina nivel cero
2^{10} páxinas de 2^{12} bytes no nivel 1

total: 4 Megabytes + 4 Kbytes

e) **El sistema operativo Linux requiere que los procesos tengan áreas separadas de código, datos y pila.**
1. Linux no puede implementarse sobre esta arquitectura porque requiere una arquitectura de 64 bits.
2. Linux no puede implementarse sobre esta arquitectura porque requiere una tabla de segmentos en memoria para cada proceso.
3. Linux se implementa sobre esta arquitectura usando los registros del procesador para indicar las direcciones base y límite de los segmentos.
4. Linux se implementa sobre esta arquitectura pero no en la forma indicada en 3. En caso de elegir esta opción debe explicar como.
Indique la respuesta correcta:_____
O sistema operativo Linux precisa que os procesos teñan áreas separadas de código, datos e pila.
1. Linux non se pode implementar nesta arquitectura porque require unha arquitectura de 64 bits.
2. Linux non se pode implementar encima desta arquitectura porque require unha táboa de segmentos de memoria para cada proceso.
3. Linux está implementado enriba desta arquitectura usando os rexistros do procesador para indicar os enderezos base e límite dos segmentos.
4. Linux está implementado nesta arquitectura pero non da forma indicada en 3. En caso de escoller esta opción, debes explicar como.
Indica a resposta correcta:_____
The Linux operating system requires processes to have separate code, data, and stack areas.
1. Linux cannot be implemented on this architecture because it requires a 64-bit architecture.
2. Linux cannot be implemented on top of this architecture because it requires a memory segment table for each process.
3. Linux is implemented on top of this architecture using the processor registers to indicate the base and limit addresses of the segments.
4. Linux is implemented on this architecture but not in the way indicated in 3. In case of choosing this option, you must explain how.
Please indicate the correct answer:_____

Correcta 4. Explicado en clase para linux 64 bits paxinación en 4 niveis. A explicación e similar adaptada a esta arquitectura.

APELLIDOS:_____ **NOMBRE:**_____ **DNI:** _____

Sistemas Operativos (PROCESOS) – Grado Ingeniería Informática UDC. Enero 2023

-Apellidos en MAYUSCULA.

-Examen sin nombre equivale a NO PRESENTADO.

Q1.-(1 punto) Responda Verdadero/Falso (o no conteste) a cada pregunta (0.1 cada una). Respuesta incorrecta -0.1. 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: 0.1;Wrong answer: -0.1 (0.1 x 8 = 0.8 points) Minimum score for this question: 0.*Cada unha das preguntas seguintes son de responder V/F (Verdadeiro/Falso) ou non responder. Resposta correcta: +0.1; Resposta incorrecta: -0.1. Puntuación mínima para esta pregunta: 0

1	2	3	4	5	6	7	8	9	10
F	**F**	**V**	**F**	**F**	**V**	**F**	**F**	**F**	**F**

1-La mayoria de los procesadores modernos tienen la instruccion "terminar proceso" en su juego de instrucciones. *Most modern processors have the "terminate process" instruction in their instruction set.* A maioría dos procesadores modernos teñen a instrución de "terminar proceso" no seu conxunto de instrucións.

2.-En un sistema monoprocesador un programa solo puede dar lugar a un proceso. *In a uniprocessor system, a program can only generate one process.* Nun sistema uniprocesador, un programa só pode dar lugar a un proceso.

3.-Si un proceso es el único proceso listo en el sistema, su prioridad no influye en el tiempo de ejecución. *If a process is the only ready process in the system, its priority has no influence on its execution time.* Se un proceso é o único proceso listo do sistema, a súa prioridade non ten influencia no tempo de execución.

4.-Salvo que las modifiquen despues de haber sido creados, dos procesos del mismo usuario tienen el mismo conjunto de variables de entorno. *Unless modified after they have been created, two processes of the same user have the same set of environment variables*. A menos que as modifiquen despois de ser creados, dous procesos do mesmo usuario teñen o mesmo conxunto de variables de ambiente.

5.-Salvo que las modifiquen despues de haber sido creados, dos procesos que ejecutan el mismo programa tienen el mismo conjunto de variables de entorno. *Unless modified after they have been created, two processes that execute the same program have the same set of environment variables*. A menos que as modifiquen despois de ser creados, dous procesos que executan o mesmo programa teñen o mesmo conxunto de variables de ambiente.

6.-Un manejador de interrupciones es siempre parte del código del kernel. *An interrupt handler is always part of the kernel code*. Un manexador de interrupcións é siempre parte do código do kernel.

7.-Un proceso con la credencial real de usuario del root puede acceder a cualquier fichero en el sistema. *A process with the real user credential of root can access any file on the system.* Un proceso coa credencial real de usuario de root pode acceder a calquera ficheiro do sistema.

8.-En un sistema round-robin de cuanto 100ms, si un proceso tienen una ráfaga CPUde 80ms, el siguiente puede obtener la CPU durante 120ms seguidos. *In a 100ms quantum round-robin system, if one process has a CPU burst of 80ms, the next one may get the CPU for 120ms in a row*. Nun sistema round-robin de quantum 100, se un proceso ten unha ráfaga de CPU de 80, o seguinte pode obter a CPU 120ms seguidos.

9.-Desde que es creado, hasta que termina, un proceso no puede realizar mas de una llamada exec. *From the time it is created until it is terminated, a process cannot make more than one exec system call.* Desde o momento en que se crea ata que finaliza, un proceso non pode realizar máis dunha chamada exec.

10.-Existen situaciones en las que fork() puede cambiar la credencial. *There are situations where fork() can change the user credential.* Hai situacións nas que fork() pode cambiar a credencial

Q2.-(0.5 puntos) Sea el siguiente código en C, con todos los includes necesarios y que com-

APELLIDOS:_____ NOMBRE:_____ DNI: _____

pila correctamente. *Consider the following C code, with all the necessary includes and that compiles correctly.* Sexa o seguinte código en C, con todas os includes necesarios e que compila correctamente

```
int BuscarVariable (char * var, char *e[])
{
  int pos=0;
  char aux[MAXVAR];

  strcpy (aux,var);
  strcat (aux,"=");

  while (e[pos]!=NULL)
    if (!strncmp(e[pos],aux,strlen(aux)))
      return (pos);
    else
      pos++;
  return(-1);
}

int SV(char *v1, char *v2, char * vle, char *e[])
{
  int pos;
  char *aux;

  if ((pos=BuscarVariable(v2,e))!=-1)
    {errno=EEXIST;return(-1);}
  if ((pos=BuscarVariable(v1,e))==-1)
    {errno=ENOENT;return(-1);}

  if ((aux=(char *)malloc(strlen(v2)+strlen(vle)+2))==NULL)
    {errno=ENOMEM; return -1;}
  strcpy(aux,v2);
  strcat(aux,"=");
  strcat(aux,vle);
  e[pos]=aux;
  return (pos);
}
```

Indicar cual de las siguientes afirmaciones es CIERTA SIEMPRE (A), puede ser cierta a veces (B) o es FALSA SIMPRE (C), después de una EXITOSA ejecución (que no devuelve -1) de la funcion SV. *Please indicate which of the following statements is ALWAYS TRUE (A), can be true sometimes (B) or is ALWAYS FALSE (C), after a SUCCESSFUL execution (that does not return -1) of function SV,* . Indica cal das seguintes afirmacións é SEMPRE VERDADEIRA (A), ás veces pode ser verdadeira (B) ou SEMPRE É FALSA (C), despois dunha execución EXITOSA (que non devolve -1) da función SV. (+0.1 correcta, -0.05 incorrecta)

APELLIDOS:_____ **NOMBRE:**_____ **DNI:** _____

1	2	3	4	5
B	**C**	**C**	**C**	**B**

1.-El entorno de env (tercer argumento de main) y el de environ son el mismo. *The environment of env (third argument of main) and that of environ are the same.* O entorno de env (terceiro argumento de main) e o de environ son o mesmo.
2.-El número de variables de entorno en uno de los entornos (env o environ) ha cambiado. *The number of environment variables in one of the environments (env or environ) has changed.* O número de variables de ambiente nun dos ambientes (env o environ) cambiou.

3.-Una variable de entorno aparece duplicada en alguno de los entornos. *An environment variable appears duplicated in some of the environments.* Unha variable de ambiente aparece duplicada nalgúns dos ambientes.

4.-Aunque los entornos estén correctamente terminados a NULL y las variables sean cadenas correctamente terminadas, es posible que esta función produzca segmentation fault. *Even if the environments are properly null-terminated and the variables are properly terminated strings, it is possible for this function to produce a segmentation fault.* Aínda que os ambientes estean correctamente terminados en nulo e as variables sexan cadeas correctamente terminadas, é posible que esta función produza un fallo de segmentación.

5.-Una variable que solo existia en uno de los entornos ahora existe en los dos. *A variable that only existed in one of the environments now exists in both.* Unha variable que só existía nun dos ambientes existe agora en ambos. **EXPLICACION: La funcion SV substituye la variable v1 por v2 con valor vle (cadena "v2=vle") comprobando previamente que v2 no existe y que v1 sí existe)**

1-B: serán el mismo o no, dependiendo de si lo eran antes de llamar a SV
2-C: SV substituye una variable por otra, por tanto no cambia el número de variables
3-C: SV comprueba que v2 no existe previamente
4-C: la asignación de memoria es correcta (el +2 es para el "=" y el caracter fin de cadena). Si no hay memoria disponible la función devuelve -1
5-B: Es posible, en el caso de que la variable sustituta es la que existe en el entorno donde no ejecutamos SV. Ahora existe en los dos

Q2.-(0.5 puntos) Sea el siguiente código en C, con todos los includes necesarios y que compila correctamente y que produce un ejecutable a.out. *Consider the following C code, with all the necessary includes and that compiles correctly and the corresponding a.out executable file.* Sexa o seguinte código en C, con todas os includes necesarios e que compila correctamente e que produce un ficheiro executable a.out.

Tanto a.out como f1.txt son del usuario u1, a.out es ejecutado por un usuario u2, 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 u1, a.out is executed by user u2, 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

APELLIDOS:_____ **NOMBRE:**_____ **DNI:** _____

usuario u1, a.out executado polo usuario u2, 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_RDWR);
    df2=open ("./f1.txt", O_RDONLY);
    printf ("%d, %d\n", df1, df2);
}
```

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

EXPLICACION: *exec()* cambia la credencial efectiva a la del propietario del fichero cuando el fichero ejecutable tiene el premiso setuid, por tanto, los casos en que *a.out* tiene el permiso setuid la credencial efectiva es *u1*. Para el acceso a ficheros se comprueba la credencial efectiva: cuando la credencial efectiva es *u1* se miran los permisos de propietario, y cuando es *u2* los permisos del resto (es indistinto que *u1* y *u2* sean o no del mismo grupo: los permisos de grupo son iguales que los de resto)

APELLIDOS:_____**NOMBRE:**_____ **DNI:** _____

USE LETRAS MAYÚSCULAS EN APELLIDOS Y NOMBRE

<<**Parte E/S (1.5 puntos)**>>. Sistemas Operativos - Grado Ingeniera Informática UDC. Enero de 2023

No se admite entrega de hojas adicionales - - HAY QUE contestar en los espacios proporcionados

1: (0.5 puntos/points)

Ante la siguiente llamada a función `fprintf(fich,"%s-%d\n","feliz",2023)`, respóndase V/F (Verdadero/Falso) o deje en blanco.

Respuesta correcta: +0.1; Respuesta incorrecta: -0.1. Puntuación mínima en esta pregunta: 0.

Ante la siguiente llamada a función fprintf(fich , "%s-%d\n", "feliz", 2023), responda V/F (Verdadeiro/Falso) ou deixe en branco.

Resposta correcta: +0.1; Resposta incorrecta: -0.1. Puntuación mínima nesta pregunta: 0.

Given the following call fprintf(fich , "%s-%d\n", "feliz", 2023), answer T/F (True/False) or leave as blank.

Correct answer: +0.1; Wrong answer: -0.1. Minimum score for this question: 0.

A	B	C	D	E
F	F	F	V	F

← Responde V/F aquí, o deja en blanco

Responde V/F aquí, ou deixa en branco

either answer T/F here, or leave blank

A- La capa del subsistema de E/S denominada "software independiente del dispositivo" prepara un **buffer** en el que se almacena la cadena "feliz-2023\n" y a continuación realiza una llamada al sistema.

*A capa do subsistema de E/S chamada "software independente do dispositivo" prepara un **buffer** no que se almacena a cadea "feliz-2023\n" e a continuación realiza unha chamada ao sistema.*

*The layer of the I/O subsystem named "Device Independent Software" creates a **buffer** where the string "feliz-2023\n" is stored, and then makes a system call.*

B- Dado el **buffer** del apartado A, internamente **fprintf** realizará una llamada a write(1, **buffer**, 11). Asúmase que no se ha hecho ninguna llamada a **close** antes de la llamada a **fprintf**.

*Dado o **buffer** do apartado A, internamente **fprintf** realizará unha chamda a write (1, **buffer**, 11). Asúmase que non se fixo ningunha chamada a **close** antes da chamada a **fprintf**.*

*Given the **buffer** of the previous Section A, **fprintf** internally makes a call to write (1, **buffer**, 11). Let's assume no calls to **close** where done before the call to **fprintf**.*

C- La capa del subsistema de E/S denominada "controlador de dispositivo o device driver" se encarga de calcular qué bloque (o bloques) de disco debe(n) estar en memoria para completar la operación write.

A capa do subsistema de E/S denominada "controlador de dispositivo ou device driver" encárgase de calcular cal é o bloque (ou bloques) de disco que debe(n) estar en memoria para completar a operación write.

The layer of the I/O subsystem named "device driver" is in charge of computing which is (are) the block (or blocks) from disk that must be within memory to complete the write operation.

D- Si el fichero asociado ocupaba 4090 bytes antes de llamar a fprintf, y el sistema de ficheros que lo contiene usa bloques de 4096 bytes, la capa del subsistema de E/S denominada "software independiente del dispositivo" hará que se asigne un nuevo bloque al fichero. Dicho bloque estará en la caché de bloques y a él se copiarán los datos "2023\n". El nuevo tamaño del fichero será de 4101 bytes.

Se o ficheiro asociado ocupaba 4090 bytes antes de chamar a fprintf, e o sistema de ficheiros que o contén usa bloques de 4096 bytes, a capa do subsistema de E/S denominada "software independente de dispositivo" fará que se asigne un novo bloque ao ficheiro. Dito bloque estará na caché de bloques e a él serán copiados os datos "2023\n". O novo tamaño do ficheiro será de 4101 bytes.

If the associated file occupied 4090 bytes before the call to fprintf, and the filesystem where that file is located uses 4096-byte blocks, the layer of the I/O subsystem named "device independent software" will assign a new block to that file. Such a block will be in the blocks cache, and the data "2023\n" will be copied into it. The new size of the associated file will be of 4101 bytes.

E- En algún momento, la capa del subsistema de E/S denominada "controlador de dispositivo o device driver" lanzará una interrupción para que los dos bloques asociados al fichero sean transferidos desde la caché de bloques a disco.

Nalgún momento, a capa do subsistema de E/S denominada "controlador de dispositivo ou device driver" lanzará unha interrupción para que os dous bloques asociados ao ficheiro sexan transferidos desde a caché de bloques a disco.

At some moment, the layer of the I/O subsystem called "device driver" will raise an interruption to ensure that the two blocks associated to the file will be transferred from the blocks cache to disk.

2: (0.5 puntos/points)

La cola de peticiones de E/S para acceder a cilindros de un disco contenía las peticiones [185, 190, 33, 400, 250, 200]. Ya se han atendido las peticiones [200] y [190] (en este orden). ¿En qué orden se atenderán las restantes peticiones?

A cola de peticións de E/S para acceder a cilindros dun disco contén as peticións [185, 190, 33, 400, 250, 200]. Xa foron atendidas as peticións [200] e [190] (nesta orde). Indíquese en que orde serán atendidas as restantes peticións.

The I/O requests queue for accessing cylinders within a disk contained the requests [185, 190, 33, 400, 250, 200]. Requests [200] and [190] have already been attended (in this order). Indicate in which order the pending requests will be attended.

Algoritmo / *algorithm*	Cilindros accedidos / *cylinders accessed*					
SSTF	200	190	185	250	400	33
C-LOOK	200	190	185	33	400	250
CSCAN	200	190	185	33	400	250
SCAN	200	190	185	33	250	400

3: (0.5 puntos/points)

El fichero "a.dat" contiene "ABCDEFGH\n". ¿Qué se escribe en pantalla en las llamadas a printf de las líneas lin.09 y lin.17?

O ficheiro "a.dat"contén "ABCDEFGH\n". Que se escribe en pantalla nas chamadas a printf das liñas lin.09 e lin17 ?

File "a.dat" contains "ABCDEFGH\n". What is written into the screen in the calls to printf within lines lin.09 and lin.17 ?

```
//lin.01:   int  main(int argc, char *argv[])
//lin.02:   {errno=0;
//lin.03:    struct aiocb aio;     long ret1, ret2, ret3;
//lin.04:    char BUF1[8]={'\0','\0','\0','\0','\0','\0','\0','\0'};
//lin.05:    char BUF2[8]={'\0','\0','\0','\0','\0','\0','\0','\0'};
//lin.06:    int fd = open("a.dat", O_RDONLY);
//lin.07:    lseek(fd, 5, SEEK_CUR);
//lin.08:    ret1 = read(fd, BUF1, 10);
//lin.09:    printf("%s--%ld",BUF1,ret1);              /**/

//lin.10:    aio.aio_fildes  = fd;
//lin.11:    aio.aio_buf     = BUF2; aio.aio_nbytes  = 4;
//lin.12:    aio.aio_reqprio = 0;    aio.aio_offset  = 2;
//lin.13:    aio.aio_sigevent.sigev_notify = SIGEV_NONE;
//lin.14:    ret2= aio_read(&aio);
//lin.15:    while(aio_error(&aio)==EINPROGRESS);
//lin.16:    ret3 = aio_return(&aio);
//lin.17:    printf("%s--%ld",BUF2,ret2);              /**/
//lin.18:   }
```

Nota: asúmase que no hubo ningún error de ejecución.

Nota: Asúmase que non houbo erros durante a execución.

Note: Let us assume no errors were reported during the execution.

printf lin.09: =

F	G	H	\n	-	-	4	

← Contesta aquí / *fill your answer here*

Pon un char en cada celda / *write a char in each cell*

Breve justificación */ breve xustificación / brief explanation* ➔

lseek posiciona el offset de lectura en el fichero en la posición 5 → leeré a partir de la posición con la 'F'.
read trata de leer 10 bytes, pero sólo quedan 4 por leer (FGH\n) y los almacena en BUF1 → FGH\n
ret1 guarda el valor devuelto por read; esto es, cuántos bytes se han leído → 4

printf lin.17: =

C	D	E	F	-	-	0	

← Contesta aquí / *fill your answer here*

Pon un char en cada celda / *write a char in each cell*

Breve justificación */ breve xustificación / brief explanation* ➔

BUF2 contiene 4 chars del fichero, leídos a partir del offset 2 del fichero → CDEF
ret2 guarda un 0 (indica si se han producido errores en aio_read: no → 0 (-1 si hay error al encolar petición)
ret3 guarda cuántos bytes se han leído = 4
el printf muestra ret2 = 0;