Apellidos, Nombre: _____ **DNI:**_____ **Grupo:**_____

1º de Grado en Ingeniería Informática (Grupos 1, 2 y 4)

Examen final de Fundamentos de Computadores (Convocatoria de junio)

26 de mayo de 2016
(TEMAS 1, 2, 3, 4, 5 y 6)

Instrucciones para realizar el examen (tipo **A**)

- El tiempo disponible es de 3 horas.
- No olvide poner los apellidos y el nombre tanto en la hoja de examen como en los folios entregados.
- Para las preguntas tipo test, selecciónese una única respuesta para cada cuestión en la siguiente tabla (señalándola con una X). El resto de preguntas (2ª parte del examen) se contestarán en folios o en el propio enunciado (si así es indicado).
- Cada dos respuestas incorrectas en el test anulan una correcta. Una pregunta sin contestar ni suma ni resta.
- Todos los alumnos deberán entregar tanto la hoja del examen como los folios utilizados o no al acabar.

Parte I: tipo test (37,5%; 0.15 puntos por respuesta)

A	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25
a																									
b																									
c																									

T1. En relación a los puente norte y puente sur de una placa de PC:
a) El puente sur comunica dispositivos con menor ancho de banda que el puente norte.
b) La funcionalidad del puente sur tiende a ser integrada, actualmente, en la propia CPU.
c) En caso de estar presentes ambos, lo habitual es que el puente sur necesite un disipador de calor, mientras que el puente norte no.

T2. El vocablo *firmware* se refiere a:
a) El conjunto de componentes eléctricos (fuentes de alimentación, condensadores, cables, resistencias, etc.), de una computadora.
b) El conjunto de componente electrónicos (procesador, memoria, tarjetas E/S, etc.) de una computadora.
c) Ninguna de las anteriores.

T3. El número 1111 1111 1000 0000 0000 0000 0000 0000 en formato IEEE 754 representa:
a) NaN (Not a Number).
b) Menos infinito.
c) Un cierto número negativo con un valor absoluto muy grande.

T4. Sea el número $9CA1)_{16}$, expresado en C2 de 16 bits. Si quisiéramos extenderlo a 24 bits, representando el mismo valor entero, tendríamos el siguiente valor (en este caso expresado en octal):
a) $77716241)_8$
b) $00116241)_8$
c) Ninguno de los anteriores.

T5. Sean los dos números expresados en octal $34)_8$ y $17)_8$. El resultado de hacer el XOR lógico, bit a bit en posiciones correspondientes, de ambos será (expresado también en octal):
a) $13)_8$.
b) $54)_8$.
c) $23)_8$.

T6. El número obtenido al desplazar una secuencia de dígitos hexadecimales 3 posiciones a la izquierda (añadiendo ceros por la derecha) es:
a) El número original multiplicado por 16^3.
b) El número original multiplicado por 3*16.
c) El número original multiplicado por 3^{16}.

T7. El mayor entero positivo que puede representarse en 4 bytes (usando representación en complemento a 2) es:
a) 2^{32}.
b) 2^{31}.
c) 2^{31}-1.

T8. Un circuito lógico combinacional con n líneas de entrada y 2^n líneas de salida se llama:
a) Decodificador.
b) Codificador.
c) Multiplexor

T9. Indica cual de las siguientes afirmaciones es la única VERDADERA:
a) En un circuito secuencial la salida en cada instante dependerá únicamente de las entradas a dicho circuito en ese instante.
b) Una ROM de 1024 posiciones de 8 bits cada una será más compleja (esto es, tendrá más puertas lógicas) que una PLA de tamaño 10x100x8.
c) Un circuito combinacional cualquiera de cuatro entradas y una salida se podrá implementar siempre con un multiplexor de 4 a 1, con la única ayuda de un inversor (puerta NOT) adicional.

T10. Considérese una función lógica que, tras ser completamente simplificada, queda como F(A,B,C,D) = A*C+B. Podemos decir sin miedo a equivocarnos que:
a) El mayor implicante primo del mapa de Karnaugh resultante cubrirá 8 unos.
b) El mayor implicante primo del mapa de Karnaugh resultante cubrirá 4 unos.
c) El mayor implicante primo del mapa de Karnaugh resultante cubrirá 2 unos.

T11. Una memoria de sólo lectura que es borrable y reprogramable electrónicamente es una:
a) EEPROM.
b) PLA.
c) ROM.

T12. Sólo uno de los siguiente nombres de fichero es válido en Linux y se puede utilizar, por tanto, con el comando touch:
a) *examen<feb,may,jul>* que genera 3 ficheros llamados *examenfeb*, *examenmay* y *examenjul*.
b) *examenfeb;examenmay;examenjul*, que genera 3 ficheros llamados *examenfeb*, *examenmay* y *examenjul*.
c) *c:jaime* es validísimo aunque no haga referencia a una unidad c: de almacenamiento.

T13. En el terminal de Linux, sólo una de las siguientes rutas es absoluta:
a) /usr/lib, que nos dirige al directorio usr que cuelga del raíz y de éste a lib.
b) ~/usr/lib, que nos dirige al directorio usr que cuelga del raíz y de éste a lib.
c) ../usr/lib, que nos dirige siempre al directorio usr que cuelga del raíz y de éste a lib.

T14. Cuando hablamos de procesos en Linux, el redireccionamiento de éstos consiste básicamente en:
a) Intercambiar entre sí los papeles de stdin y stdout al ejecutar un sólo comando.
b) Modificar el stdin con el stdout almacenado en un fichero (o generar un fichero con el stdout) o tomar como stdin de una orden la salida stdout de otra orden.
c) Pasar parte de la carga del procesamiento de un programa en ejecución a otro procesador ocioso.

T15. Para poder leer un dato de teclado, un programa de un usuario debe:
a) Acceder directamente, mediante instrucciones, al hardware de la controladora del dispositivo.
b) Crear un proceso hijo que se encarga de dicha lectura accediendo al periférico de entrada
c) Realizar una llamada al sistema operativo (*syscall*) para que éste se encargue de dicha tarea.

T16. En Linux, cuando lanzamos varios procesos en segundo plano desde la línea de comandos (terminal), el símbolo que se utiliza es:
a) ;
b) $
c) &

T17. Los tamaños de los registros que podemos usar en el ensamblador x86-64 son de:
a) 4, 8, 16, 20 y 24 bits (Los de 4 bits ocupan la parte más baja de los registros o los 4 bits siguientes).
b) Sólo los tamaños de 32 y 64 bits (los de 8 y 16 bits quedaron ya obsoletos y no se usan más).
c) 8, 16, 32 y 64 bits básicamente.

T18. En el ensamblador del x86-64, los registros que podemos usar como operandos de uso general son:
a) RAX, RBX, RCX, RDX, R8-R15, aunque algunos de éstos puedan ser usados por el procesador como operando destino de ciertas instrucciones.
b) RAX, RBX, RCX, RDX, R8-R15, RDI, RSI, RSP y RBP son todos de propósito general, por lo que podemos usarlos sin ninguna precaución especial.
c) El programador sólo puede usar el RAX, RBX, RCX, RDX, el RSP y el RBP para uso de propósito general, el resto está dedicado para uso exclusivo del propio procesador.

T19. En la programación en ensamblador de los procesadores Intel x86-64 llamamos operandos inmediatos a:
a) Aquellos operandos que están codificados en la propia instrucción.
b) Aquellos operandos que al ejecutar la instrucción, los encuentra en la dirección de memoria inmediatamente posterior a la que apunta el registro RDI (Registro de Destino Inmediato).
c) A los operandos que se encuentran en algún registro de acceso inmediato del procesador.

T20. Un fichero objeto (módulo objeto) contiene, entre otras cosas:
a) Las librerías estáticas (*.a* en Linux) a las que llama, pero no las dinámicas (*.so* en Linux).
b) Información de reubicación (para instrucciones con accesos a memoria, saltos, etc.).
c) Siempre, en cualquier caso, el código fuente que generó dicho fichero al ser traducido.

T21. Si dispongo de la red 192.168.0.0/16 y le asigno a mis equipos la máscara 255.255.240.0 ¿cuántas subredes estoy obteniendo?:
a) 16.
b) 4096.
c) Si la red es 192.168.0.0/16, la máscara de mis subredes debe ser obligatoriamente 255.255.0.0.

T22. Con el comando *host* podemos obtener:
a) La IP asociada a un nombre de máquina dado y viceversa.
b) La configuración de red del equipo en el que se ejecute dicho comando.
c) El servidor DNS de un determinado host.

T23. En redes, las direcciones de puerto son necesarias para:
a) Poder distinguir entre varias direcciones privadas de red que tengamos dentro de una red NAT, pero b) no es cierta.
b) Poder distinguir entre varias aplicaciones de red que se ejecuten en un mismo host. pero a) no es cierta.
c) Tanto a) como b) son ciertas.

T24. Una dirección IP V4 consta de 32 bit, de los cuales:
a) Siempre los dos primeros bytes identifican la red y los restantes nuestra IP dentro de la red.
b) Parte de ellos identifican en qué red estamos y el resto cuál es nuestro host.
c) Los dos primeros bytes identifican la red y los restantes el número MAC de nuestra interfaz.

T25. En redes, DHCP significa:
a) Domain Hypertext Configuration Protocol y sirve para traducir dominios de páginas web a direcciones IP concretas.
b) Delivered Home Configuration Protocol y sirve para asignar direcciones IP V4 privadas de redes NAT.
c) Dynamic Host Configuration Protocol y sirve para configurar automáticamente parámetros de red en un host tales como máscara y direcciones IP y la IP del router.

Apellidos, Nombre: _____ **DNI:**_____ **Grupo:**_____

1º de Grado en Ingeniería Informática (Grupos 1, 2 y 4)

Examen final de Fundamentos de Computadores (Convocatoria de junio)

26 de mayo de 2016

(TEMAS 1, 2, 3, 4, 5 y 6)

Parte II: cuestiones teórico-prácticas *(30%; puntuación indicada en cada apartado)*

C1. **a)** *(0,6 puntos)* Sabiendo que sigue el formato IEEE 754 de doble precisión, determinar el número real que se corresponde con la ristra de 64 bits 0x408F450000000000, almacenada según el convenio *big-endian*.

b) *(0,6 puntos)* Indicad los 4 bytes (expresados en hexadecimal, con sus respectivos desplazamientos resultantes) que habría que almacenar a partir de un desplazamiento 0000:001C de un archivo para escribir el número +200,125 en punto flotante en formato IEEE 754 de 32 bits, siguiendo un convenio de almacenamiento *little-endian*.

C2. *(0,6 puntos) Las redes NAT (Acrónimo de* **(C2A)** *) son típicas de entornos domésticos. Internamente utilizan rangos de direcciones IP privadas como por ejemplo* **(C2B)** *ó* **(C2C)**, *mientras que para el mundo exterior se traducen (típicamente) a una única dirección IP de tipo* **(C2D)**. *Para ello disponen de un router para realizar la conversión entre ambas direcciones, externa e interna. Así, si un host de la red interna desea enviar una conexión al exterior, se comunica con su router donde en los paquetes TCP/UDP envía los valores de su propia dirección IP y puerto de origen, dirección IP destino y puerto de destino. El router, al reenviar el paquete al exterior cambiará la IP* **(C2E)** *por la dirección IP* **(C2F)** *y el número de puerto* **(C2G)** *por un valor diferente que elige. Por cada conexión así conformada va creando una tabla denominada* **(C2H)** *que es precisamente la que consultará cuando reciba los paquetes de respuesta para reenviarlos internamente al host y en los que en la cabecera debe reponer la IP y el puerto* **(C2I)** *por la IP y puerto originales que encuentra en la tabla gracias a la coincidencia con el valor que guardó de* **(C2J)**.

C3. *(0,6 puntos) Esta pregunta trata sobre la pila en el x86-64.*

La pila es una zona de la **(C3A)** *principal que se maneja de una manera especial. La pila es una estructura de datos en la que existe un puntero para conocer la "cima de la pila", o posición de memoria donde se almacenará el siguiente dato a apilar. La pila, en el procesador x86-64, crece (aumenta su contenido) desde direcciones* **(C3B)** *a direcciones* **(C3C)** *de la memoria principal (a esta operación se le llama "apilar" y decrece (se sacan datos de ella o "desapila") hacia direcciones* **(C3D)**. *Se dice que es una estructura tipo* **(C3E)**. *El puntero o registro donde se almacena el valor de la cima de la pila es el* **(C3F)**, *mientras que el* **(C3G)** *se apunta a los marcos de pila de las distintas funciones, que contienen sus respectivos parámetros, variables locales, direcciones de retorno, etc..*

Las siguientes instrucciones son algunos ejemplos de instrucciones que hacen un uso implícito de la pila: **(C3H)** *o* **(C3I)**, *mientras que otras sirven para manejar la pila de forma explícita, como por ejemplo* **(C3J)**.

C4. *(0,6 puntos) En la instrucción de ensamblador x86-64 mov var, %ecx el operando var se encuentra en* **(C4A)**, *y se accede al mismo (direccionamiento) de forma* **(C4B)**. *En cambio, en la instrucción mov (%rax),%rcx el operando señalado por (%rax) se accede de forma* **(C4C)**, *encontrándose en %rax* la **(C4D)** *de memoria donde se encuentra dicho operando.*

En la instrucción *mov %rbx, %rax* el operando *%rbx* se encuentra en uno de los **(C4E)** del procesador; sin embargo, en la instrucción *mov $3100, %ebx* el operando *$3100* viene en la propia **(C4F)** de forma **(C4G)**.

Por otra parte, en lo que respecta a categorías o tipos de instrucciones: las instrucciones *add, shr, and etc.* pertenecen al tipo de instrucciones denominadas **(C4H)**, *mov* es una instrucción del tipo de movimiento de datos, *jmp* es una instrucción de tipo **(C4I)**, y *jle* lo es de tipo **(C4J)**.

Parte III: ejercicios boletines *(32,5%; puntuación indicada en cada apartado)*

P1. *(0,7 puntos)* Dado el siguiente pantallazo de *okteta*, correspondiente al INICIO de un archivo (es decir, no se muestran más que las primeras líneas del mismo, no el archivo completo):

```
0000:0000 50 36 0A 23 20 41 72 63 68 69 76 6F 20 70 61 72 P6.# Archivo par
0000:0010 63 69 61 6C 0A 37 35 20 31 30 30 0A 32 35 35 0A cial.75 100.255.
0000:0020 00 00 FF 00 00 FF 00 00 FF 00 00 FF 00 00 FF 00 ..ÿ..ÿ..ÿ..ÿ..ÿ.
0000:0030 00 FF 00 00 FF 00 00 FF 00 00 FF 00 00 FF 00 00 .ÿ..ÿ..ÿ..ÿ..ÿ..
```

a) Indicad a qué tipo de archivo se corresponde, así como todos los datos relevantes que puedas determinar a partir de los bytes que aparecen en su cabecera. En particular, ¿qué representa en dicha cabecera cada vez que aparece el byte `0x0A`?

b) Determinad el tamaño en bytes exacto de dicho archivo (OJO, porque no aparece entero en el pantallazo, pero aún así su tamaño puede determinarse exactamente).

P2. *(1 punto)* Dada la función expresada en su forma normal canónica $f(A_3, A_2, A_1, A_0) = \sum m(0, 2, 6, 8, 14, 15) + d(10, 11, 12\ 13)$ (*)*, se pide:*

a) Representadla en el mapa de Karnaugh correspondiente.

b) Localizad sobre el anterior mapa de Karnaugh todos los implicantes primos esenciales, indicando cuál o cuáles minitérminos, para cada implicante primo, lo hacen esencial.

c) Localizad todos los implicantes primos no esenciales usados en la cobertura, indicando porqué son "no esenciales".

d) Realizad una cobertura mínima en el mapa de Karnaugh para implementar la función como suma de productos.

e) Expresad algebraicamente esa función lógica mínima como suma de productos.

(*) "**+ *d (10,11,12,13)*"** significa que son minitérminos tipo "no importa" (indeterminados).

P3. *(0,8 puntos) Rellenar el siguiente texto (indicando en el examen la correspondiente referencia al hueco para cada respuesta):*

El comando **(P3A)** *sirve para terminar una sesión interactiva de línea de comandos. Si quiero saber qué procesos se están ejecutando en un instante dado, deberé ejecutar el comando* **(P3B)**, *pero el comando* **(P3C)** *es más útil para ir monitorizando los mismos en tiempo real. Mi directorio actual lo puedo saber ejecutando el comando* **(P3D)**. *Si ejecuto el comando* `cd -`, *acabaré en el directorio* **(P3E)**, *mientras que si estoy en mi directorio de inicio y ejecuto el comando* `cd ./../../boot` *acabaré en el directorio cuya ruta absoluta es* **(P3F)**. *Si deseo consultar en el terminal todas las opciones disponibles para el comando* `date`, *deberé ejecutar el comando* **(P3G)**. *Los ficheros de configuración de la mayoría de programas del sistema de una instalación Linux se suelen encontrar en el directorio* **(P3H)** *(que cuelga directamente de la raíz), mientras que los directorios* `/bin` *y* `/usr/bin` *suelen contener* **(P3I)** *y el directorio* `/sbin` **(P3J)**. *Por su parte, los subdirectorios de datos de los usuarios cuelgan todos del directorio común* **(P3K)**, *mientras que el directorio* **(P3L)** *es usado por los distintos programas para guardar archivos temporales. Finalmente, para encontrar todos los directorios que cuelgan (recursivamente) del subdirectorio* `lib` *(ubicado dentro del subdirectorio* `usr`, *éste último ya colgando directamente de la raíz), y cuyo nombre acaba con una* `z` *o una Z, el comando completo a (con todas las opciones pertinentes) a utilizar será* **(P3M)**.

P4. *(0,75 puntos)* Aprovechando las pasadas fiestas de primavera, te has creado una red privada en tu casa. La has diseñado para que dicha red cubra el ordenador de tus padres, el tuyo, el de una hermana, y una impresora en red. Tu proveedor de servicios de Internet te ha dicho que vuestro router tiene la dirección pública 203.2.46.38, que tiene servicio de NAT y que para configurar una red privada uses las direcciones 192.168.0.0. Procurando utilizar el mínimo rango de IPs (tamaño de subred lo más pequeño posible), se pide:

a) ¿Cuántos bits tendrá tu red en el "Host Id" de la máscara para configurarla?

b) Dirección de la red privada.

c) Dirección de broadcast de la misma.

d) Dirección privada del router (la primera posible).

e) Direcciones privadas de los equipos.

Soluciones:

C1.

a) N=1000.625

b) Ristra de 32 bits = 0x43482000 → Bytes 00, 20, 48 y 43 en desplazamientos 0000:001C, 0000:001D, 0000:001E y 0000:001F, respectivamente.

C2. Redes NAT:

C2A: Network Address Translation.
C2B y C2C: 10 · 0 · 0 · 0 / 8 | 172 · 16 · 0 · 0 / 12 | 192 · 168 · 0 · 0 / 16 | 169 · 254 · 0 · 0 / 16.
C2D: Pública o Externa.
C2E: Privada | NAT | del Host | de origen.
C2F: del ROUTER | Pública | Externa.
C2G: de origen | del Host | de origen de la conexión.
C2H: Tabla NAT |Tabla de traducciones NAT. | Tabla de traducciones
C2I: de destino | internos de destino.
C2J: Nº de Puerto asignado | Nº de Puerto asignado al Host | Nº de puerto sustituido | Nº de puerto sustituido por el nº de puerto origen | ...

***C3.* Esta pregunta trata sobre la pila en el x86-64...**

C3A: memoria.
C3B: altas
C3C: bajas.
C3D: altas.
C3E: LIFO
C3F: %rsp | %sp | rsp | sp.
C3G: %rbp | %bp | rbp | bp.
C3H: CALL | RET | LEAVE | INT | IRET
C3I: CALL | RET | LEAVE | INT | IRET.
C3J: Push | POP | MOV | ...

C4. En la instrucción de ensamblador x86-64 mov var, %ecx

C4A: memoria.
C4B: directa.
C4C: indirecta.
C4D: dirección.
C4E: registros.
C4F: instrucción.
C4G: directa | inmediata.
C4H: aritmético-lógicas.
C4I: salto incondicional.
C4J: salto condicional.

P1. Dado el siguiente pantallazo de okteta...

a) Se trata de un archivo de imagen de tipo PPM. Se puede determinar a primera vista al ver sus dos primeros bytes (caracteres "P6") que son el número mágico de este tipo de archivos. A continuación vienen una serie de caracteres ASCII separados por retornos de carro (que es el significado concreto del byte 0x0A), que indican, por este orden:

- "# Archivo parcial" → comentario sobre el archivo (se sabe que es un comentario por empezar con "#").
- "75 100" → Indican el número de filas y de columnas de la imagen.
- "255" → Indican el máximo valor que puede tomar cada píxel en su color de cada canal R, G, B. En este caso, es una profundidad de 8 bits, lo que provoca un valor de color para cada canal en el intervalo [0..255].

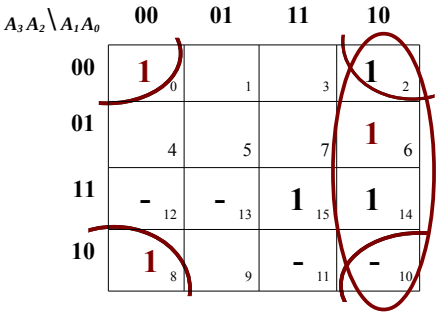
Ya a partir del desplazamiento 0000:0020 comienza la masa de bytes en sí, a razón de tres bytes por cada píxel.

b) El tamaño exacto de la imagen será 32 bytes (tamaño cabecera) + 3*100*75 (tamaño mapa de píxeles) = 22532 bytes.

P2. Dada la función expresada en su forma normal canónica...
a) Representadla en el mapa de Karnaugh correspondiente.

$A_3A_2 \backslash A_1A_0$	00	01	11	10
00	1 (0)	(1)	(3)	1 (2)
01	(4)	(5)	(7)	1 (6)
11	- (12)	- (13)	1 (15)	1 (14)
10	1 (8)	(9)	- (11)	- (10)

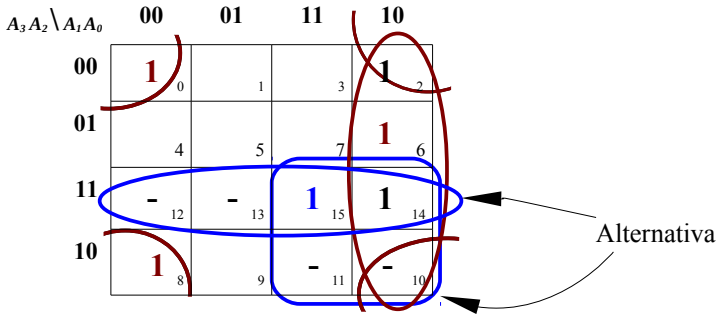
***b)* Localizad sobre el anterior mapa de Karnaugh todos los implicantes primos esenciales, indicando cuál o cuáles minitérminos, para cada implicante primo, lo hacen esencial.**



Los minitérminos 0 y 8 sólo pueden ser cubiertos por un único implicante primo de orden 2 (0, 2, 8, 10).

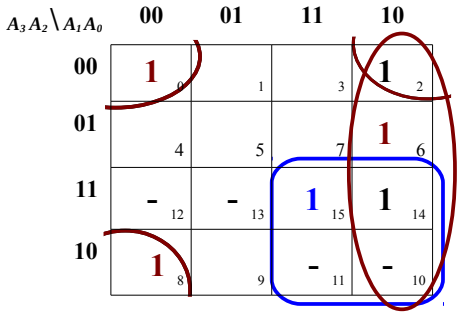
El minitérmino 6 sólo pueden ser cubierto por un único implicante primo de orden 2 (2, 6, 14, 10).

c) Localizad todos los implicantes primos no esenciales usados en la cobertura, indicando porqué son no esenciales.



El minitérmino 15 puede ser cubierto por el implicante primo (orden 2) (15, 14, 11, 10) ó también por el implicante (también de orden 2) (12, 13, 15, 14), luego la elección de uno cualquiera de los dos nos llevaría a una implementación mínima. Luego son implicantes primos "no esenciales".

d) Realizar una cobertura mínima en el mapa de Karnaugh para implementar la función como suma de productos.



Hay varias coberturas, pero todas ellas necesitan, obligatoriamente, contener los implicantes primos esenciales (2, 6, 14, 10) y (0, 2, 8, 10), y para cubrir el minitérmino 15 elegimos o bien el implicante primo no esencial (12, 13, 15, 14) o bien el (15, 14, 11, 10).

e) ***Expresad esa función lógica mínima como suma de productos.***

$f(A_3, A_2, A_1, A_0) = \overline{A_2} \cdot \overline{A_0} + A_1 \cdot \overline{A_0} + A_3 \cdot A_1$ (Por ejemplo).

$f(A_3, A_2, A_1, A_0) = \overline{A_2} \cdot \overline{A_0} + A_1 \cdot \overline{A_0} + A_3 \cdot A_2$ (Alternativa).

P3. Rellenar el siguiente texto...

P3A: exit (o pulsar Ctrl-D)...
P3B: ps.
P3C: top.
P3D: pwd.
P3E: último en el que estuve (anterior al actual).
P3F: /boot.
P3G: man date.
P3H: /etc.
P3I: comandos y otros programas.
P3J: comandos de administrador.
P3K: /home.
P3L: /tmp.
P3M: find /usr/lib -iname "*z"

P4. **Aprovechando las pasadas fiestas de primavera...**

a) 3
b) 192.168.0.0.
c) 192.168.0.7.
d) 192.168.0.1.
e) De la 192.168.0.2 a la 192.168.0.5 (Por ejemplo).

TODO Versión **A**

A	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	T18	T19	T20	T21	T22	T23	T24	T25
a	X			X		X		X		X	X		X					X	X		X	X			
b			X						X					X						X				X	
c		X			X		X					X			X	X	X						X		X