Se dispone de una partición de 256GB de un disco duro que da soporte a un sistema de ficheros UNIX con las siguientes características:

- Los bloques lógicos del sistema de ficheros son de 4KB.
- El sistema de ficheros utiliza i-nodos con 7 punteros directos, un indirecto simple, uno doble y uno triple. Además la cabecera del i-nodo (permisos, fechas de modificación, tipo, etc.) ocupan 24 bytes.
- Como el tamaño medio de un fichero es menor de 4KB, se crea el sistema de ficheros con un ratio 1 a 1 entre i-nodos y bloques de datos.

a) Considerando el tamaño de la partición, ¿qué tamaño en bytes parece razonable utilizar para almacenar un puntero a bloque?

b) Si consideramos que el superbloque ocupa un solo bloque lógico (al que se le da el número 1) ¿indique dónde comienza (número de bloque) y de qué tamaño es cada una de las estructuras del sistema de ficheros en la partición?

Estructura	**Comienzo**	**Fin**	**Tamaño**
…	…	…	…
…	…	…	…

Si el disco duro que alberga a la participación sufre algún tipo de daño que hace que determinados bloques de disco no se puedan leer, cuál es el efecto que se produciría si:

i. el error impide leer los bloques del 7500 al 9100 de la partición.

ii. el error impide leer el 25% final de la partición.

Responda a las siguientes cuestiones, indicando si habría alguna diferencia entre los casos i) y ii) antes citados:

a) ¿Se podría leer la estructura de directorios completa (no los contenidos de los ficheros, sólo la estructura de directorios)? ¿Por qué?

b) ¿Sabríamos si un bloque de datos o un i-nodo está ocupado? ¿Y a quién pertenece?

c) Imagínese que uno de los ficheros contenía una imagen en formato no comprimido (un bitmap en 256 colores) que ocupaba varios megabytes. ¿Cómo podríamos recuperar el contenido del fichero? ¿valdría para algo saber las dimensiones de la imagen en número de pixels de ancho y alto?

d) Si uno de los ficheros afectados es un fichero que contiene un certificado digital (menor de 2KB con una cabecera de un formato determinado) ¿sería posible recuperar el certificado?

Una vez recuperado el contenido o eliminado aquel que no pueda ser identificado, ¿de qué forma se podría modificar el diseño o el uso del sistema de ficheros de esta partición para poder usarlo sin que los bloques dañados afecten al uso futuro del fichero? Desarrolle este diseño respondiendo a las siguientes cuestiones:

e) ¿Dónde se podría guardar el rango o rangos de bloques que no son utilizables? Considere que los errores de este tipo se suelen producir en bloques de disco contiguos, son poco frecuentes y afectan a un número del orden de un centenar a varios miles de bloques.

f) ¿Es razonable marcar en los bitmaps esos bloques de una forma especial? Considere la posibilidad de marcarlos como libres, como ocupados o con un tipo de código diferente. ¿Cuál es mejor solución? Razone la respuesta por simplicidad y por consumo de recursos.

Respuestas:

a) 256 x 2^{30} bytes / 4 x 2^{10} bytes/bloque = 2^{26} bloques ➔ Parece razonable usar 32 bits (que permiten direccionar 2^{32} bloques), por lo tanto 4 bytes por dirección de bloque.

b) Tamaño del i-nodo= 24 bytes de cabecera + 4 bytes/dirección-bloque x (7 punteros directos + 1 simple + 1 doble + 1 triple) = 64 bytes.

Número de i-nodos por bloque= 4 x 2^{10} bytes/bloque / 64 bytes/i-nodo = 2^{12} / 2^{6} i-nodos/bloque = 2^{6} i-nodos/bloque = 64 i-nodos/bloque

Si el disco tiene (aproximadamente), en bloques de datos 2^{26} bloques y considerando el ratio 1:1 i-nodos bloques, se necesitarán (aproximadamente también) 2^{26} i-nodos, que ocupan 2^{26} i-nodos / 2^{6} i-nodos/bloque = 2^{20} bloques de i-nodos.

Si calculamos el tamaño de los bitmaps en bloques (para ambos elementos, i-nodos y bloques de datos) tenemos que tener un bit por elemento, habiendo (aproximadamente) 2^{26} elementos ➔ necesitamos= 2^{26} bits / (4 x 2^{10} bytes/bloque x 2^{3} bits/byte) = 2^{26} / 2^{15} bloques = 2^{11} bloques = 2048 bloques.

Así pues, el layout del disco queda como:

Estructura	**Comienzo**	**Fin**	**Tamaño**
Superbloque	1	1	1
Bitmaps i-nodos	2	2049	2048
Bitmaps bloques	2050	4097	2048
Vector i-nodos	4098	4097+2^{20}	2^{20}
Vector bloques	4098+2^{20}	2^{26}	~2^{26}

Los bloques de disco que han fallado son, para el caso i) los correspondientes al vector de i-nodos y para el caso ii) son únicamente bloques de datos.

c) En ninguno de los casos, un directorio es un i-nodo y uno o varios bloques de datos, de forma que si faltan cualquiera de ambas estructuras, no es posible recorrer la estructura de directorios.

d) Sí se podría saber la ocupación o no de los elementos, siempre y cuando los bitmaps no se vean afectados. Sobre la pertenencia a un fichero o a otro de cada recurso (i-nodo o bloque de datos) nos encontramos con la siguiente casuística:

i. Los i-nodos están referenciados siempre por bloques de datos, en concreto bloques de datos de directorio. Por lo tanto podríamos saber para determinados i-nodos a qué fichero pertenecían. El problema se plantea en el momento en el cual alguna de esas entradas sea un directorio. Al no poderse acceder al i-nodo no se sabe cuáles son los bloques de contenidos.

ii. Si se han perdido bloques de datos, desde el i-nodo podemos identificar parte de ellos, en base a lo que son los punteros directos. Pero los punteros indirectos (de cualquier tipo) hacen referencia a los bloques de datos por medio de bloques intermedios de indexación. Si éstos se han dañado, se pierden todas las direcciones de los bloque indexados por ellos.

e) Asumimos que la imagen no tiene cabecera, y que es un formato raster crudo (es decir que empezando por la coordenada 0,0, y avanzando por filas el fichero contiene el color e cada pixel. A 256 colores, estamos hablando de un byte por pixel (8 bits, 2^8=256 colores). Si conocemos las dimensiones de la imagen podemos saber cuántos bloques de datos de tamaño tiene el fichero y, más importante, qué pixels representa un bloque en concreto del fichero (de acuerdo con su posición relativa dentro del fichero). Analizamos, de nuevo, los dos casos:

   i. Si se ha perdido el i-nodo del fichero (en ese rango es la única parte del fichero que puede verse afectada), no sabríamos qué bloque va en cada posición del fichero, ni cuáles son bloques de contenido (de la imagen) ni cuáles son bloques de datos asociados al direccionamiento intermedio de los punteros indirectos. Si además hay varios ficheros afectados, los bloques de datos que sabemos que están ocupados (lo indica el bitmap) pueden ser del fichero que nos interesa (datos o bloques de punteros) o de otros ficheros. Como mucho se podrían tener muchos ficheros recuperados, cada uno de un bloque de disco con cada uno de los bloques de datos ocupados (pero no se podrían combinar entre ellos, al menos con la información que se dispone).

   ii. Si se han perdido bloques de datos, sí se podría recuperar parte de la imagen, desde luego, la parte que se corresponde con los bloques de acceso directo desde el i-nodo (si estos no están dañados), o los que se puedan acceder por los indirectos. Para los que no se pueda acceder se puede dejar esa área de la imagen vacía (en un color determinado), y los bloques que aparecen ocupados y no están enlazados desde un i-nodo (alguno de los bloques de indirección intermedios se ha dañado), se podrían recuperar como ficheros de un bloque (sin identificar dónde van esos pixel en la imagen).

f) En este caso nos encontramos con una situación diferente, por dos motivos, por un lado, el fichero es suficientemente pequeño para ser indexado desde un puntero directo desde el i-nodo (en realidad con el primer puntero es más que suficiente), además si el fichero tiene una cabecera característica, que puede terne un formato más o menos conocido, un número mágico o similar, de forma que se puede comprobar para unos datos desconocidos si aquello “se parece” a la cabecera de un certificado. Teniendo en cuenta estas características:

i. Si se ha perdido el i-nodo podemos hacer una búsqueda exhaustiva entre los bloques ocupados y no referenciados por un i-nodo no dañado, para ver si el comienzo del bloque se corresponde con esa cabecera. Sí podríamos recuperarlo.

ii. Si se ha dañado un bloque de datos que contenía el certificado, ahí sí que no se puede hacer nada.

g) Esta información podría guardarse en el superbloque, como metadatos asociados al sistema de fichero. Con las características que tiene sería suficiente como un par de valores, que indicasen el primer bloque dañado, y cuántos consecutivos están en ese estado. Si se producen más daños, se incluiría otro par de valores para la nueva región dañada (salvo que se puedan describir con un solo par de valores (sean regiones contiguas).

h) Sería razonable marcar los bloques dañados en el bitmap. Principalmente para que el algoritmo de localización de nuevos recursos, no los intente usar. Una vez decidido que puede ser interesante la cuestión es si es conveniente etiquetar el elemento como ocupado o con otra etiqueta diferente. Si se usa otra etiqueta tendríamos que un elemento puede estar, libre, ocupado o dañado, tres estados diferentes, por lo tanto necesitaríamos más de un bit por posición, no es una solución inabordable, pero desde luego que afecta a toda la algorítmica de gestión del bitmap.

Si en lugar de eso se marcan sólo como ocupados, el problema lo tendríamos al hacer una verificación de la integridad del sistema de ficheros, tendríamos, por ejemplo bloques ocupados que nadie referencia o viceversa (con los i-nodos). Pero este problema se puede resolver si usamos los rangos de valores de bloques dañados que incluíamos en el superbloque. De esta forma el chkdsk (utilidad para la verificación la integridad del sistema de ficheros), se saltaría los elementos que (independientemente de que vengan marcados como ocupados) están en los rangos dañados. Ese cambio sólo afectaría a dicha herramienta, así que el impacto en consumo de recursos (bitmaps del mismo tamaño) y el impacto en el diseño (modificaciones mínimas, y sólo en una herramienta auxiliar), aconsejan utilizar esta segunda alternativa.