Mini Teste

Sistemas de Operação

Ano Lectivo de 2017/2018

29/10/2017

NºMec: _____ **Nome:** _____

Na resolução deste exame, tenha em consideração o seguinte:

- **As questões são independentes entre si.** Assim, a resposta a qualquer questão deve considerar o estado do disco tal como apresentado e não aquele que resultaria da execução do código apresentado numa outra qualquer questão.
- **As questões devem ser respondidas no contexto concreto do disco apresentado.**
- Pode responder às questões pela ordem que quiser, muito embora se responder primeiro à primeira questão ganhará uma compreensão do sistema de ficheiros que o ajudará na resposta às restantes.
- As questões 1, 7 e as três mais bem cotadas das restantes têm cotação de 3 valores cada; as outras têm cotação de 2,5 valores cada.
- A duração do exame é de 75+15mn.
- À saída, **deve entregar** tudo o que recebeu.

Considere que se criou um disco virtual sobre um ficheiro *linux*. Esse disco foi formatado usando o programa `mksofs` e montado no diretório `/tmp/mnt/`, usando o programa `sofsmount`. Diversas operações de manipulação de ficheiros (ficheiros regulares, diretórios e atalhos) foram a seguir efetuadas sobre esse diretório (ponto de montagem).

As listagens das páginas 5 a 8 representam o estado interno de alguns blocos/clusters do disco após as operações anteriores, mostrados usando a ferramenta `showblock`. Alguns campos do superbloco e do nó-i número 1 da tabela de nós-i foram intencionalmente substituídos por `???`. A tabela de nós-i é apenas parcialmente mostrada; todos os nós-i não mostrados estão livres e limpos e não foram alterados desde a formatação. Os campos `atime`, `mtime` e `ctime` não são mostrados. Para facilitar a leitura, nos campos `name` das entradas de directório o carácter `'\0'` foi substituído por um espaço.

---

1. Complete o preenchimento da tabela seguinte com a informação referente a todos os ficheiros não apagados residentes no disco.

caminho absoluto (***path***)	**nº do nó-i** (***nInode***)	**tipo** (**dir/file/symlink**)	**refcount**
/	0		

2. Nos dados apresentados abaixo sobre o estado das estruturas de dados internas do sistema de ficheiros, alguns campos foram intencionalmente substituídos por `???`.

   (a) Apresente os valores dos seguintes campos do superbloco.

   `ntotal`: ..... `itsize`: .....

   `ctotal`: ..... `cfree`: .....

   (b) Apresente o valor do seguinte campo do nó-i (*inode*) número 1.

   `inode[1].next`: ..... `inode[1].prev`: .....

   ⋮
   ⋮

---

3. Considere que o excerto de código seguinte é executado, não tendo sido gerada nenhuma exceção.

```
uint32_t n = soAllocInode(S_IFREG);
```

   (a) Apresente os valores, após a execução do excerto de código, dos campos do superbloco indicados abaixo, assim como o valor da variável `n`.

   `itotal`: ..... `ifree`: .....

   `ihead`: ..... `n`: .....

   (b) Há outros nós-i (*inodes*), além daquele cujo número foi armazenado na variável `n`, alterados em consequência da execução? Se sim, quais os nós-i alterados e que alterações sofreram? Se não, indique porquê?

   ⋮
   ⋮

---

4. Considere que o excerto de código seguinte é executado, não tendo sido gerada nenhuma exceção.

```
int ih = iOpen(3);
soFreeFileClusters(ih, 10);
soFreeFileClusters(ih, 3);
```

   (a) Apresente os valores, após a execução do excerto de código, dos campos do superbloco seguintes, sendo que `ref[*]` representa todo o array útil. Pode usar notação compactada, quando aplicável. Se não respondeu à questão 2, considere que antes da execução `cfree = 1000`.

   `cfree`: .....

   rcache: `idx`: ..... `ref[*]`: .....

   icache: `idx`: ..... `ref[*]`: .....

   (b) Apresente os valores dos seguintes campos do nó-i número 3, sendo que `d[*]` representa o array na totalidade.

   `size`: ..... `clucnt`: .....

   `d[*]`: .....

   `i1`: ..... `i2`: .....

(c) Há um cluster da zona de dados alterado em consequência da execução do excerto de código anterior. Qual? Que alterações sofre?

⋮
⋮
⋮

---

5. Considere que o excerto de código seguinte é executado, não tendo sido gerada nenhuma exceção.

```
int ih0 = iOpen(0);
uint32_t n1 = soGetDirEntry(ih0, "aaaa");
int ih1 = iOpen(n1);
uint32_t n2 = soDeleteDirEntry(ih1, "gggg");
```

(a) Que valores são armazenados nas variáveis `n1` e `n2`?

`n1`: .... `n2`: ....

(b) Apresente os valores, após a execução do excerto de código, dos campos do superbloco seguintes. Se não respondeu à questão 2, considere que antes da execução `cfree = 1000`.

`ifree`: .... `cfree`: ....

(c) Dos 3 clusters de dados com entradas de diretório apresentados neste exame, qual ou quais sofrem alterações em consequência da execução do excerto de código? Que alterações sofrem?

⋮
⋮
⋮
⋮
⋮

---

6. Considere que o excerto de código seguinte é executado, e que após a sua execução `ret` tem o valor 0.

```
#define PERM 0755
int ret = soMkdir("/zzzz", PERM);
```

(a) Que campos dos nós-i em uso ou que ficaram em uso após a execução sofrem alterações em consequência da execução do excerto de código? Que alterações sofrem?

(b) Que clusters da zona de dados sofrem alterações em consequência da execução do excerto de código? Que alterações sofrem?

---

7. O comando `rm -rf dir`, sendo `dir` um directório, apaga-o, incluindo todos os ficheiros, atalhos e outros diretórios nele contidos. Considere que, sendo o diretório corrente o ponto de montagem do disco, executa o comando seguinte

```
rm -rf "dddd"
```

À frente de cada uma das seguintes chamadas de sistema do sofs16 coloque um `sim` ou um `não` para respetivamente indicar se ela é invocada no contexto de execução do comando anterior. Se não sabe, deixe em branco, porque uma resposta errada desconta um valor igual ao que conta uma resposta correta.

`soLink`:	`soUnlink`:	`soMknod`: .....
`soRename`:	`soRead`:	`soWrite`: .....
`soTruncate`:	`soReaddir`:	`soMkdir`: .....
`soRmdir`:	`soSymlink`:	`soReadlink`: .....

Estado da estrutura de dados interna do disco

Block 0 as superblock data

```
   Magic number: 0x50F5
   Version number: 0x2017
   Volume name: mini-teste-modelo
   Properly unmounted: yes
   Number of mounts: 0
   Total number of blocks in the device: ???
Inode table metadata:
   First block of the inode table: 1
   Number of blocks of the inode table: ???
   Total number of inodes: 16
   Number of free inodes: 8
   Head of list of free inodes: 10
Free cluster table metadata:
   First block of the free cluster table: 3
   Number of blocks of the free cluster table: 1
   Index of first byte to retrieve references: 6
Clusters metadata:
   First block of the cluster zone: 4
   Total number of clusters: ???
   Number of free clusters: ???
   Retrieval cache:
      Index of the first filled cache element:  23
      Cache contents:
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) 24 25 26 27 28 29 30
         31 32 33 34 35 36 37 38 39 40
         41 42 43 44 45 46 47 48 49 50
         51 52 53
   Insertion cache:
      Index of the first free cache element:  5
      Cache contents:
         16 17 23 5 6 (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil) (nil)
         (nil) (nil) (nil)
```

Block 1 as inode entries

```
Inode #0
type =  directory, permissions = rwxrwxr-x, lnkcnt = 4, owner = 1000, group = 1000
size in bytes = 2048, size in clusters = 1
d[] = {0 (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #1
type = free regular file, permissions = rw-rw-r--, lnkcnt = 0, owner = 1000, group = 1000
size in bytes = 4000, size in clusters = 0
next = ???, prev = ???
d[] = {(nil) (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #2
type =  directory, permissions = rwxrwxr-x, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = 2048, size in clusters = 1
d[] = {1 (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #3
type =  regular file, permissions = rw-rw-r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 42000, size in clusters = 7
d[] = {11 (nil) (nil) (nil) 12 13}, i1 = 14, i2 = (nil)
----------------
Inode #4
type =  regular file, permissions = rw-rw-r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 10000, size in clusters = 5
```

```
d[] = {2 3 4 21 22 (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #5
type =  directory, permissions = rwxrwxr-x, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = 2048, size in clusters = 1
d[] = {7 (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #6
type =  regular file, permissions = rw-rw-r--, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = 4000, size in clusters = 2
d[] = {8 9 (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #7
type =  regular file, permissions = rw-rw-r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 12, size in clusters = 1
d[] = {18 (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
```

Block 2 as inode entries

```
Inode #0
type =  symlink, permissions = rwxrwxrwx, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 12, size in clusters = 1
d[] = {10 (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #1
type = free regular file, permissions = rw-rw-r--, lnkcnt = 0, owner = 1000, group = 1000
size in bytes = 4, size in clusters = 0
next = 1, prev = 15
d[] = {(nil) (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
Inode #2
type = free clean, permissions = ----------, lnkcnt = 0, owner = 0, group = 0
size in bytes = 0, size in clusters = 0
next = 11, prev = 1
d[] = {(nil) (nil) (nil) (nil) (nil) (nil)}, i1 = (nil), i2 = (nil)
----------------
    ...
```

Block 3 as a bitmap block

```
cnt = 55, idx = 6
0000: 00 00 00 00 00 00 03 ff ff ff ff ff ff f8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    ...
01c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Blocks 4-7 as direntries

```
.                                                   0000000000
..                                                  0000000000
                                                    0000000001
dddd                                                0000000002
aaaa                                                0000000005
bbbb                                                0000000006
                                                    (nil)
    ...
                                                    (nil)
```

Blocks 8-11 as direntries

```
.                                                   0000000002
..                                                  0000000000
gggg                                                0000000003
ffff                                                0000000004
                                                    0000000007
aaaa                                                0000000008
                                                    (nil)
    ...
                                                    (nil)
```

Blocks 32-35 as direntries

```
.                                                   0000000005
..                                                  0000000000
gggg                                                0000000007
ffff                                                0000000006
                                                    0000000009
                                                    (nil)
    ...
                                                    (nil)
```

Blocks 60-63 as references

```
0000: 0000000015   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
0008:   (nil)      (nil)      (nil)      (nil)      (nil)    0000000019 0000000020   (nil)
0016:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
0024:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
    ...
0504:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
```

Blocks 44-47 as ascii

```
0000: .  .  /  a  a  a  a  /  f  f  f  f  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0032: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    ...
2016: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Blocks 76-79 as ascii

```
0000: .  .  /  d  d  d  d  /  f  f  f  f  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0032: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
    ...
2016: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Assinatura das funções referenciadas neste exame

```
int iOpen(uint32_t in);
uint32_t soAllocInode(uint32_t type);
void soFreeFileClusters(int ih, uint32_t ffcn);
uint32_t soGetDirEntry(int pih, const char *name);
uint32 soDeleteDirEntry(int pih, const char *name);
int soMkdir(const char *path, mode_t mode);
```

Declaração das estruturas de dados internas do sofs16

```
#define ReferenceBytesPerBitmapBlock (BlockSize - 4) /* reference bytes per bitmap block */

struct SORefBlock /* Definition of the reference bitmap block data type. */
{
    uint16_t cnt;                              /* number of references (bits at one) in block */
    uint16_t idx;                              /* index of first non-empty byte in block */
    uint8_t map[ReferenceBytesPerBitmapBlock]; /* the bit map in the block */
};
```

```
#define REFERENCE_CACHE_SIZE  53  /* size of reference cache */

struct SOReferenceCache {
    /** \brief storage area whose elements are logical numbers of free clusters */
    uint32_t ref[REFERENCE_CACHE_SIZE];  ///< the cache area
    uint32_t idx;                        ///< retrieval/insertion point
};

struct SOSuperBlock
{
    uint16_t magic; /* magic number - file system identification number */
    uint16_t version; /* version number */

    char name[PARTITION_NAME_SIZE + 1]; /* volume name */

    uint8_t mntstat; /* mount status (1: properly unmounted; 0: otherwise */
    uint8_t mntcnt; /* number of mounts since last file system check */

    uint32_t ntotal; /* total number of blocks in the device */

    uint32_t itstart; /* physical number of the block where the inode table starts */
    uint32_t itsize; /* number of blocks that the table of inodes comprises */
    uint32_t itotal; /* total number of inodes */
    uint32_t ifree; /* number of free inodes */
    uint32_t ihead; /* head of linked list of free inodes */

    uint32_t rmstart; /* physical number of the block where the bitmap starts */
    uint32_t rmsize; /* number of blocks that the map of cluster references comprises */
    uint32_t rmidx; /* index of first byte to be used to retrieve references from map */

    uint32_t czstart; /* physical number of the block where the cluster zone starts */
    uint32_t ctotal; /* total number of clusters */
    uint32_t cfree; /* number of free clusters */

    SOReferenceCache rcache; /* retrieval cache of references to free clusters */
    SOReferenceCache icache; /* insert cache of references to free clusters */
};
```

```
#define SOFS17_MAX_NAME 59 /* maximum length of a file name (in characters) */

struct SODirEntry
{
    uint32_t in;                    /* the inode number */
    char name[SOFS17_MAX_NAME + 1]; /* the name of a file (NULL-terminated string) */
};
```

```
#define INODE_FREE (0001000)    /* flag signaling inode is free (it uses the sticky bit) */

#define N_DIRECT 6              /* number of direct references in the inode */

struct SOInode
{
    uint16_t mode;              /* file type and permissions */
    uint16_t lnkcnt;            /* number of hard links (directory entries) pointing to the inode */
    uint32_t owner;             /* user ID of the file owner */
    uint32_t group;             /* group ID of the file owner */
    uint32_t size;              /* file size in bytes: */
    uint32_t clucnt;            /* cluster count: total number of clusters used by the file */

    uint32_t atime;             /* time of last access to file contents */
    union
    {                           /* usage depends on inode state */
        uint32_t mtime;         /* time of last change to file contents (if inode is in use) */
        uint32_t next;          /* next free inode (if inode is free) */
    };

    union
    {                           /* usage depends on inode state */
        uint32_t ctime;         /* time of last change to inode information (if inode is in use) */
        uint32_t prev;          /* previous free inode (if inode is free) */
    };

    uint32_t d[N_DIRECT];       /* direct references */
    uint32_t i1;                /* reference to the cluster that extend the d array */
    uint32_t i2;                /* reference to the cluster that extends the i1 array */
};
```

```
#define InodesPerBlock (BlockSize/sizeof(SOInode)) /* number of inodes per block */

#define ReferencesPerBlock (BlockSize/sizeof(uint32_t)) /* number of references per block */

#define ReferencesPerCluster (ClusterSize/sizeof(uint32_t)) /* number of references per cluster */

#define ReferencesPerBitmapBlock (8*ReferenceBytesPerBitmapBlock) /* number of references per bitmap block */

#define BlocksPerCluster 4 /* number of blocks per cluster */

#define ClusterSize (BlocksPerCluster * BlockSize) /* number of bytes per cluster */

#define DirentriesPerCluster (ClusterSize/sizeof(SODirEntry)) /* number of direntries per cluster */

#define NullReference 0xFFFFFFFF /* null reference */
```