Fundamentos de Sistemas Operativos / Sistemas de Operação

Mini Teste - modelo **novembro de 2020**

Ano Lectivo de 2020/2021

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.** Respostas do tipo *se ..., então ...* não são consideradas.
- 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 nas respostas às restantes.
- As questões 1, 6 e as duas mais bem cotadas das restantes têm cotação de 3,5 valores cada; as restantes têm cotação de 3,0 valores cada.
- A duração do exame é de 60+15mn.
- À saída, **deve entregar** tudo o que recebeu (enunciado, folhas de resposta e rascunhos).

Considere que se criou um disco virtual sobre um ficheiro. Esse disco foi formatado como `sofs20`, 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 9 representam o estado interno de alguns blocos do disco após as operações anteriores, mostrados usando a ferramenta `showblock`. Alguns campos do superbloco e de dois *inodes* foram intencionalmente substituídos por `???`. A tabela de *inodes* é apenas parcialmente mostrada; todos os *inodes* não mostrados estão livres e limpos. 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. Há blocos apenas parcialmente mostrados. A parte omissa não é necessária para a resposta a qualquer questão. O mesmo acontece com os blocos não mostrados.

---

1. Complete o preenchimento da tabela seguinte com a informação referente a todos os ficheiros (representados por um caminho absoluto) não apagados residentes no disco.

caminho absoluto (*path*)	**nº do *inode* (*nInode*)**	**tipo (dir/file/symlink)**	**lnkcnt**
/	0	dir	4
/bbbb	6	file	2
/gggg	2	dir	2
/aaaa	5	dir	2
/gggg/aaaa	3	file	1
/gggg/cccc	7	file	1
/gggg/bbbb	8	symlink	1
/aaaa/gggg	6	file	2
/aaaa/ffff	4	file	1

---

2. Nos dados apresentados 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. Sabe-se que há 17 blocos de dados com referências de blocos de dados livres.

   `ntotal`: 4400 ..... `itotal`: 32 .....

   `dbtotal`: 4380 ..... `dbfree`: 4311+(68-61)+38 = 4356 .....

   (b) Apresente os valores dos seguintes campos dos *inodes* 6 e 3.

   `inode[6].blkcnt`: 2 ..... `inode[3].blkcnt`: 2+(1+6)+(1+3) = 13 .....

   (c) Apresente os valores mínimos dos seguintes campos dos *inodes* 6 e 3.

   `inode[6].size`: 1024+1 = 1025 ..... `inode[3].size`: sz1 = (4+256+6)*1024+1 .....

---

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

```
uint16_t n = soAllocInode(S_IFREG | 0644);
```

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

   `ifree`: 23 ..... `iidx`: 10 ..... `n`: 10 .....

   (b) Há outros campos do superbloco, além dos contemplados na alínea anterior, que sofrem alterações em consequência da execução do excerto de código. Qual(is) e que alteração(ões) sofre(m)? ⋮

   ⋮

   ibitmap[0] passa de 1FD para 5FD

   ⋮

   ... 0000 0001 1111 1110 | ... 0000 0100 0000 0000 = ... 0000 0101 1111 1110 = 5FD

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

```
int ih = soOpenInode(3);
soFreeFileBlocks(ih, 12);
```

(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, se aplicável. Se não respondeu à questão 2, considere que antes da execução `dbfree = 1000`. Resposta do tipo *não foi alterado* não será considerada.

`dbfree`: 4356+7 (ou 1000+7, se se usar o 1000)

`reftable.blk_idx`: 0 `reftable.ref_idx`: 0 `reftable.count`: 4311

`retrieval_cache.idx`: 61 `retrieval_cache.ref[*]`: nil ... nil 62 ...

`insertion_cache.idx`: 45 `insertion_cache.ref[*]`: ... 1 2 19 20 21 23 24 25 22 nil

(b) Apresente os valores, após a execução do excerto de código, dos seguintes campos do *inode* número 3, sendo que `d[*]`, `i1[*]` e `i2[*]` representam os arrays na totalidade. Pode usar notação compactada, se aplicável.
Se não respondeu à questão 2, considere que antes da execução `size = blkcnt = 1000`.

`size`: sz1 = (4+256+6)*1024+1 (porque o ffb não mexe no size) `blkcnt`: 13-7 = 6

`d[*]`: nil nil 11 12 `i1[*]`: 14 nil nil `i2[*]`: nil

(c) Há bloco(s) de dados alterado(s) em consequência da execução do excerto de código anterior. Qual(is)? Que alterações sofre(m)?

bloco de dados 14 é alterado

nil nil ... 15 16 17 nil
nil nil ... nil
...

bloco 22 pode ser alterado
ficando todo com nil

---

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

```
int ih0 = soOpenInode(0);
uint16_t n1 = soGetDirEntry(ih0, "aaaa");
int ih1 = soOpenInode(n1);
uint16_t n2 = soDeleteDirEntry(ih1, "gggg");
```

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

`n1`: 5 `n2`: 6

(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 `dbfree = 1000`.

`ifree`: 24 `dbfree`: 4356

(c) Dos 3 blocos 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?

Bloco de dados 4 (7 do disco):

.	5
..	0
ffff	4

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 *inodes* em uso ou que ficaram em uso após a execução sofrem alterações em consequência da execução do excerto de código? Que alterações sofrem? Não considere os campos `atime`, `mtime`, `ctime`, `owner` e `group`.

é criado um dir na raiz

o novo dir tem um bloco
de dados com as entrads
. e ..

inode 0:
lnkcnt: 4 -> 5
size: 320 -> 384

inode 10:
mode: -> S_IFDIR | 0755
lnkcnt -> 2
size: -> 2*64 = 128
blkcnt: -> 1
d[0]: -> 62

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

bloco de dados 0 (3 do disco)

linha 6 vai passar a ter a entrada
zzzz 10

bloco de dados 62 (65 do disco)

. 10
.. 0

Estado da estrutura de dados interna do disco

Disk block 0 as superblock data

ifd = 010111111101

```
Header:
  Magic number: 0x50f5
  Version number: 0x20
  Volume name: "sofs20_disk"
  Properly unmounted: no
  Number of mounts: 1
  Total number of blocks in the device: ???
Inodes' metadata:
  Total number of inodes: ???
  Number of free inodes: 24
  Last allocated inode: 9
  Inode allocation bitmap:
    0x000001fd 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
    0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
Data blocks' metadata:
  First block of the data block pool: 3
  Total number of data blocks: ???
  Number of free data blocks: ???
Reference table's metadata:
  First block of the reference table: 4383
  Number of blocks of the reference table: 17
  Index of first block with references: 0
  Index of first cell, within first block, with references: 0
  Number of references in reference table: 4311
Retrieval cache:
  Index of the first occupied cache position: 61
  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) (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)    62    63    64    65    66    67    68
Insertion cache:
  Index of the first empty cache position: 38
  Cache contents:
     13 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 54 55
     56 57 58 59 60 26 1 2 (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)
```

19,20,21,23,24,25,,22

Disk block 1 as inode entries

```
Inode #0
type = directory, permissions = rwxr-xr-x, lnkcnt = 4, owner = 1000, group = 1000
size in bytes = 320, block count = 1
 d[*] = 0 (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #1
type = free, permissions = ---------, lnkcnt = 0, owner = 0, group = 0
size in bytes = 0, block count = 0
 d[*] = (nil) (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #2
type = directory, permissions = rwxr-xr-x, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = 320, block count = 1
 d[*] = 3 (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #3
type = regular file, permissions = rw-r--r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = ???, block count = ???
 d[*] = (nil) (nil) 11 12,  i1[*] = 14 22 (nil),  i2[*] = (nil)
----------------
Inode #4
type = regular file, permissions = rw-r--r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 4096, block count = 4
 d[*] = 5 6 7 61,  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #5
type = directory, permissions = rwxr-xr-x, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = 256, block count = 1                                    0-1023
 d[*] = 4 (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil) 1024-2047
----------------
Inode #6
type = regular file, permissions = rw-r--r--, lnkcnt = 2, owner = 1000, group = 1000
size in bytes = ???, block count = ???
 d[*] = 8 9 (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #7
type = regular file, permissions = rw-r--r--, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 12, block count = 1
 d[*] = 18 (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #8
type = symlink, permissions = rwxrwxrwx, lnkcnt = 1, owner = 1000, group = 1000
size in bytes = 12, block count = 1
 d[*] = 10 (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #9
type = free, permissions = ---------, lnkcnt = 0, owner = 0, group = 0
size in bytes = 0, block count = 0
 d[*] = (nil) (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
Inode #10
type = free, permissions = ---------, lnkcnt = 0, owner = 0, group = 0
size in bytes = 0, block count = 0
 d[*] = (nil) (nil) (nil) (nil),  i1[*] = (nil) (nil) (nil),  i2[*] = (nil)
----------------
```

Disk block 3 as direntries

```
.                                                    0000000000
..                                                   0000000000
bbbb                                                 0000000006
gggg                                                 0000000002
aaaa                                                 0000000005
                                                     0000000000
        ⋮
                                                     0000000000
```

Disk block 6 as direntries

```
.                                                    0000000002
..                                                   0000000000
aaaa                                                 0000000003
cccc                                                 0000000007
bbbb                                                 0000000008
                                                     0000000000
        ⋮
                                                     0000000000
```

Disk block 7 as direntries

```
.                                                    0000000005
..                                                   0000000000
gggg                                                 0000000006
ffff                                                 0000000004
                                                     0000000000
        ⋮
                                                     0000000000
```

Disk block 17 as references

```
0000:   (nil)      (nil)      (nil)      (nil)    0000000015 0000000016 0000000017   (nil)
0008:   (nil)      (nil)      (nil)      (nil)      (nil)    0000000019 0000000020 0000000021
0016:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
0024:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
        ⋮
0248:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
```

Disk block 25 as references

```
0000:   (nil)      (nil)      (nil)      (nil)    0000000023 0000000024 0000000025   (nil)
0008:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
0016:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
        ⋮
0248:   (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)      (nil)
```

Disk block 13 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
0064: 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
        ⋮
0992: 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
```

Disk block 21 as ASCII

```
0000: .  .  /  g  g  g  g  /  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
0064: 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
        ⋮
0992: 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 soOpenInode(uint16_t in);
uint16_t soAllocInode(uint32_t mode);
void soFreeFileBlocks(int ih, uint32_t ffcn);
uint16_t soGetDirentry(int pih, const char *name);
uint16_t soDeleteDirentry(int pih, const char *name);
int soMkdir(const char *path, mode_t mode);
```

Declaração das estruturas de dados internas do sofs20

```
#define BlockSize 1024U                         /** block size (in bytes) */
#define IPB (BlockSize / sizeof(SOInode))       /** number of inodes per block (16) */
#define DPB (BlockSize / sizeof(SODirEntry))    /** number of direntries per block (16) */
#define RPB (BlockSize / sizeof (uint32_t))     /** number of references per block (256) */
#define BlockNullReference 0xFFFFFFFF           /** null reference to a data block */
#define InodeNullReference 0xFFFF               /** null reference to an inode */
```

```
#define PARTITION_NAME_LEN 19     /** maximum length of volume name */
#define REF_CACHE_SIZE 68         /** size of caches in superblock for inode references */
#define MAX_INODES (100*32)       /** size of caches in superblock for inode references */

struct SOSuperBlock               /** Definition of the superblock data type.  */
{
    uint16_t magic;               /** magic number - file system identification number */
    uint8_t version;              /** version number */
    uint8_t mntstat;              /** mount status (1: properly unmounted; 0: otherwise) */

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

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

    uint32_t itotal;              /** total number of inodes */
    uint32_t ifree;               /** number of free inodes */
    uint32_t iidx;                /** number of last allocated inode */
    uint32_t ibitmap[MAX_INODES/32] /** bitmap representing inode allocation states */

    uint32_t dbp_start;           /** physical number of the block where the data zone starts */
    uint32_t dbotal;              /** total number of data blocks */
    uint32_t dbfree;              /** number of free blocks in data zone */

    uint32_t rt_start;            /** number of the disk block where the reference table starts */
    uint32_t rt_size;             /** number of blocks the reference table comprises */

    struct ReferenceTable /** The reference table control structure */
    {
        uint32_t blk_idx; /** index, within the reference table, of the first block with references */
        uint32_t ref_idx; /** index of first cell with references, within the previous block */
        uint32_t count;   /** total number of not null references in the reference table */
    };

    ReferenceTable reftable; /** The reference table control structure */

    struct ReferenceCache /** cache of references to free data blocks */
    {
        uint32_t idx;                 /** index of first free/occupied cell */
        uint32_t ref[REF_CACHE_SIZE]; /** the cache itself */
    };

    ReferenceCache retrieval_cache; /** retrieval cache of references to free data blocks */
    ReferenceCache insertion_cache; /** insertion cache of references to free data blocks */
};
```

```
#define SOFS20_FILENAME_LEN 61             /** maximum length of a file name (in characters) */

struct SODirEntry                          /** Definition of the directory entry data type. */
{
    uint16_t in;                           /** the associated inode number */
    char name[SOFS20_FILENAME_LEN + 1];    /** the name of a file (NULL-terminated string) */
};
```

```
#define N_DIRECT 4                  /** number of direct block references in the inode */
#define N_INDIRECT 3                /** number of indirect block references in the inode */
#define N_DOUBLE_INDIRECT 1         /** number of double indirect block references in the inode */

struct SOInode                      /** Definition of the inode data type. */
{
    uint16_t mode;                  /** inode mode: it stores the file type and permissions.  */
    uint16_t lnkcnt;                /** link count: number of 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 blkcnt;                /** block count: total number of blocks used by the file */

    uint32_t atime;                  /** time of last access to file information */
    uint32_t mtime;                  /** time of last change to file information */
    uint32_t ctime;                  /** time of last change to inode information */

    uint32_t d[N_DIRECT];            /** direct references to the first data blocks with file's data */
    uint32_t i1[N_INDIRECT];         /** references to blocks that extend the \c d array */
    uint32_t i2[N_DOUBLE_INDIRECT];  /** references to a block that extends the \c i1 array */
};
```