# Algunas estructuras del Sistema de archivos

struct PARTITION {

char drive\_status;

char chs\_begin[3];

char partition\_type;

char chs\_end[3];

int lba;

int secs\_partition;

}

// Lo que vamos a escribir en el primer sector del disco

// Debe medir 512 bytes

struct MBR {

char bootstrap\_code[446];

struct PARTITION partition[4];

short boot\_signature;

}

// printf("%d\n",sizeof(struct MBR));

// Debe medir 512 bytes

struct SECBOOTPART {

char jump[4];

char nombre\_particion[8];

// Tabla de parámetros del bios

// Están los datos sobre el formato de la partición

unsigned short sec\_inicpart; // 1 sector

unsigned char sec\_res; // 1 sector reservado para el sector de boot de la partición

unsigned char sec\_mapa\_bits\_area\_nodos\_i;// 1 sector

unsigned char sec\_mapa\_bits\_bloques; // 6 sectores

unsigned short sec\_tabla\_nodos\_i; // 3 sectores

unsigned int sec\_log\_particion; // 43199 sectores

unsigned char sec\_x\_bloque; // 2 sectores por bloque

unsigned char heads; // 8 superficies

unsigned char cyls; // 200 cilindros

unsigned char secfis; // 27 sectores por track

char restante[484]; // Código de arranque

};

// printf("%d\n",sizeof(struct SECBOOT));

// Debe medir 64 bytes, importante es que el tamaño sea potencia de 2

struct INODE {

char name[18];

unsigned int datetimecreat; // 32 bits

unsigned int datetimemodif; // 32 bits

unsigned int datetimelaacc; // 32 bits

unsigned short uid; // 16 bits

unsigned short gid; // 16 bits

unsigned short perms; // 16 bits

unsigned int size; // 32 bits

unsigned short direct\_blocks[10]; // 10 x 16 bits = 20 bytes

unsigned short indirect; // 16 bits

unsigned short indirect2; // 16 bits

};

// printf("%d\n",sizeof(struct INODE));