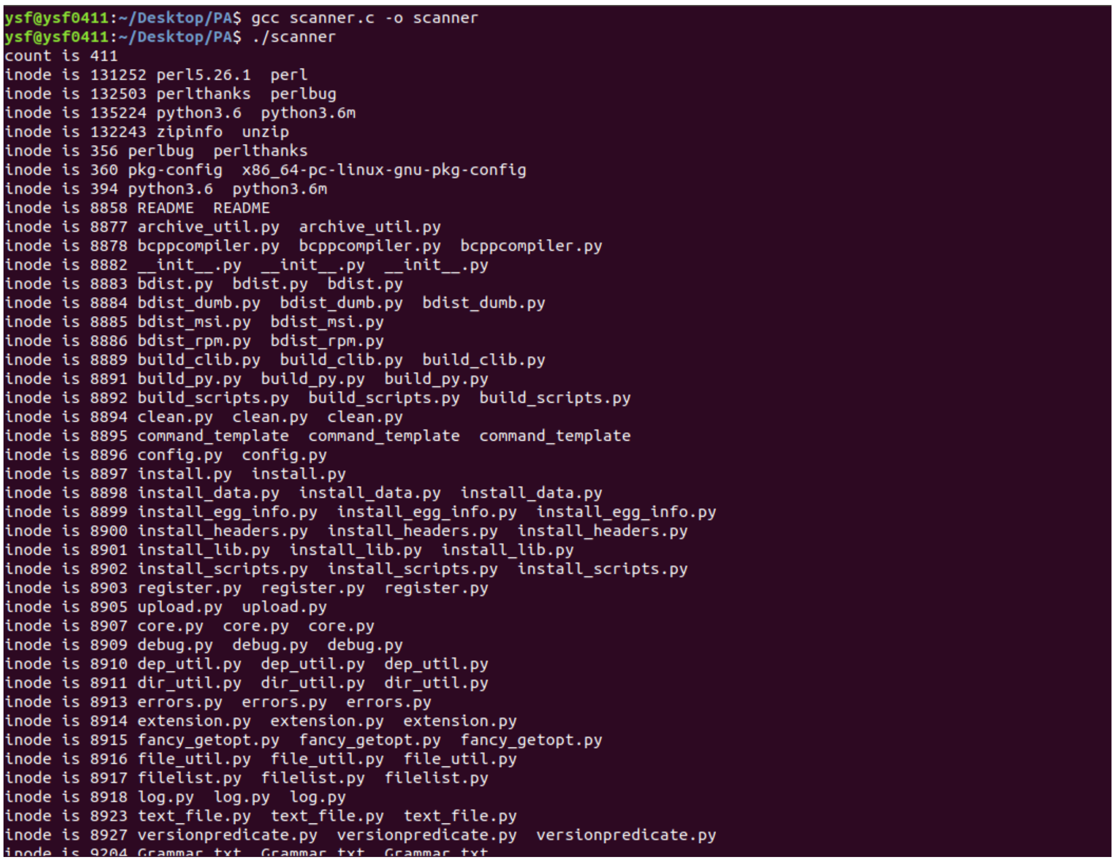
编程作业实验报告

191300073 杨斯凡 191300073@smail.nju.edu.cn

一、实验结果:

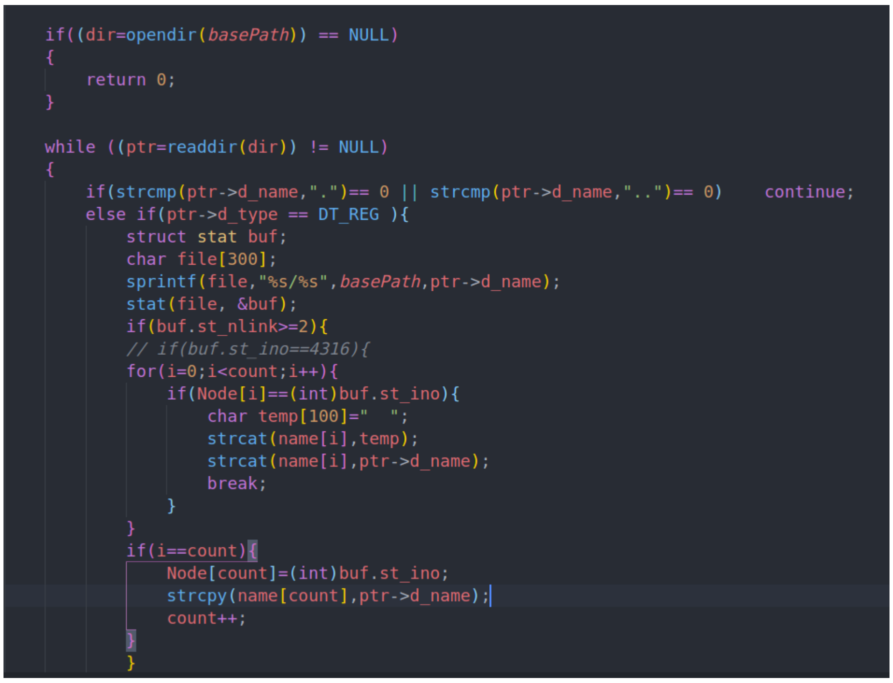


```
ysf@ysf0411:~/Desktop/PA$ gcc scanner.c -o scanner
ysf@ysf0411:~/Desktop/PA$ ./scanner
count is 411
inode is 131252 perl5.26.1  perl
inode is 132503 perlthanks  perlbug
inode is 135224 python3.6  python3.6m
inode is 132243 zipinfo  unzip
inode is 356 perlbug  perlthanks
inode is 360 pkg-config  x86_64-pc-linux-gnu-pkg-config
inode is 394 python3.6  python3.6m
inode is 8858 README  README
inode is 8877 archive_util.py  archive_util.py
inode is 8878 bcppcompiler.py  bcppcompiler.py  bcppcompiler.py
inode is 8882 __init__.py  __init__.py  __init__.py
inode is 8883 bdist.py  bdist.py  bdist.py
inode is 8884 bdist_dumb.py  bdist_dumb.py  bdist_dumb.py
inode is 8885 bdist_msi.py  bdist_msi.py
inode is 8886 bdist_rpm.py  bdist_rpm.py
inode is 8889 build_clib.py  build_clib.py  build_clib.py
inode is 8891 build_py.py  build_py.py  build_py.py
inode is 8892 build_scripts.py  build_scripts.py  build_scripts.py
inode is 8894 clean.py  clean.py  clean.py
inode is 8895 command_template  command_template  command_template
inode is 8896 config.py  config.py
inode is 8897 install.py  install.py
inode is 8898 install_data.py  install_data.py  install_data.py
inode is 8899 install_egg_info.py  install_egg_info.py  install_egg_info.py
inode is 8900 install_headers.py  install_headers.py  install_headers.py
inode is 8901 install_lib.py  install_lib.py  install_lib.py
inode is 8902 install_scripts.py  install_scripts.py  install_scripts.py
inode is 8903 register.py  register.py  register.py
inode is 8905 upload.py  upload.py
inode is 8907 core.py  core.py  core.py
inode is 8909 debug.py  debug.py  debug.py
inode is 8910 dep_util.py  dep_util.py  dep_util.py
inode is 8911 dir_util.py  dir_util.py  dir_util.py
inode is 8913 errors.py  errors.py  errors.py
inode is 8914 extension.py  extension.py  extension.py
inode is 8915 fancy_getopt.py  fancy_getopt.py  fancy_getopt.py
inode is 8916 file_util.py  file_util.py  file_util.py
inode is 8917 filelist.py  filelist.py  filelist.py
inode is 8918 log.py  log.py  log.py
inode is 8923 text_file.py  text_file.py  text_file.py
inode is 8927 versionpredicate.py  versionpredicate.py  versionpredicate.py
inode is 9204 Grammar.txt  Grammar.txt  Grammar.txt
```

(scanner)

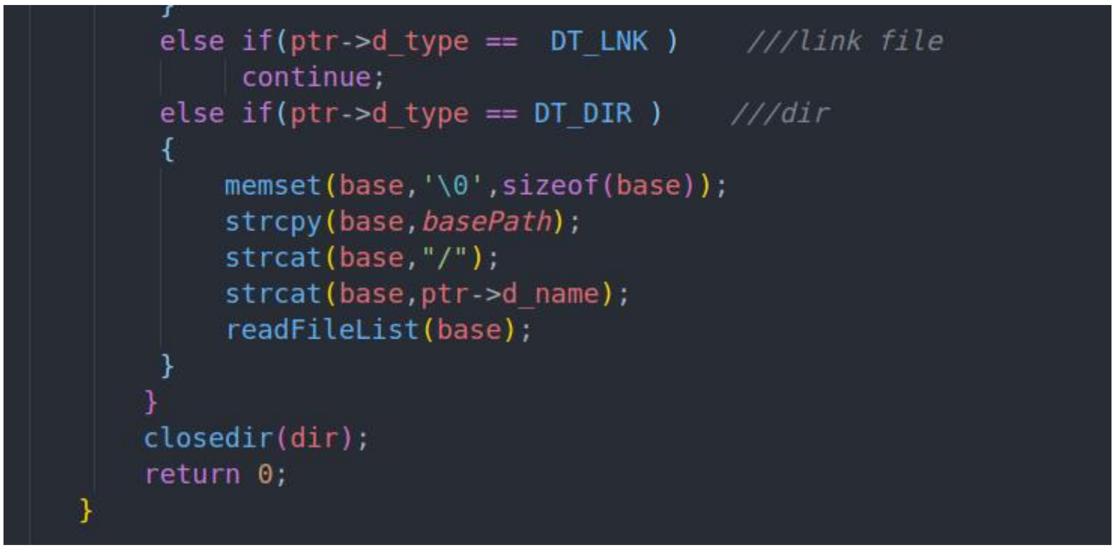
二、代码讲解

1. scanner



```c
if((dir=opendir(basePath)) == NULL)
{
    return 0;
}

while ((ptr=readdir(dir)) != NULL)
{
    if(strcmp(ptr->d_name,".")== 0 || strcmp(ptr->d_name,"..")== 0)    continue;
    else if(ptr->d_type == DT_REG ){
        struct stat buf;
        char file[300];
        sprintf(file,"%s/%s",basePath,ptr->d_name);
        stat(file, &buf);
        if(buf.st_nlink>=2){
        // if(buf.st_ino==4316){
        for(i=0;i<count;i++){
            if(Node[i]==(int)buf.st_ino){
                char temp[100]="  ";
                strcat(name[i],temp);
                strcat(name[i],ptr->d_name);
                break;
            }
        }
        if(i==count){
            Node[count]=(int)buf.st_ino;
            strcpy(name[count],ptr->d_name);
            count++;
        }
        }
```

我使用了 opendir 函数来打开目录，readdir 来进行读这个目录，读到目录下的文件之后，使用了 stat 这个 API 得到这个文件的 inode 信息，判断连接数是否大于 1，如果大于 1 就将该文件名加入对应的字符数组中，如果不是文件就递归向下读取即可.

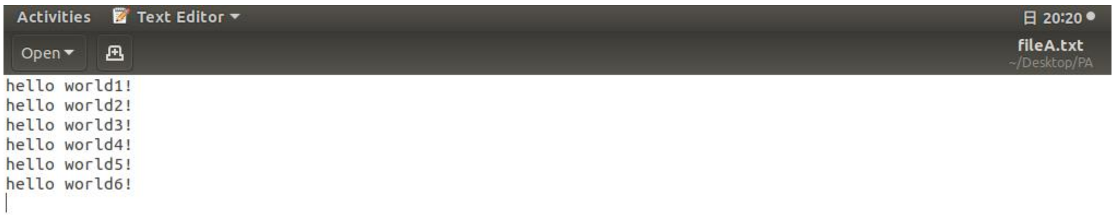


```
        }
        else if(ptr->d_type ==  DT_LNK )    ///link file
            continue;
        else if(ptr->d_type == DT_DIR )    ///dir
        {
            memset(base,'\0',sizeof(base));
            strcpy(base,basePath);
            strcat(base,"/");
            strcat(base,ptr->d_name);
            readFileList(base);
        }
    }
    closedir(dir);
    return 0;
}
```

最后输出所有的字符数组即可.

2. FAT12

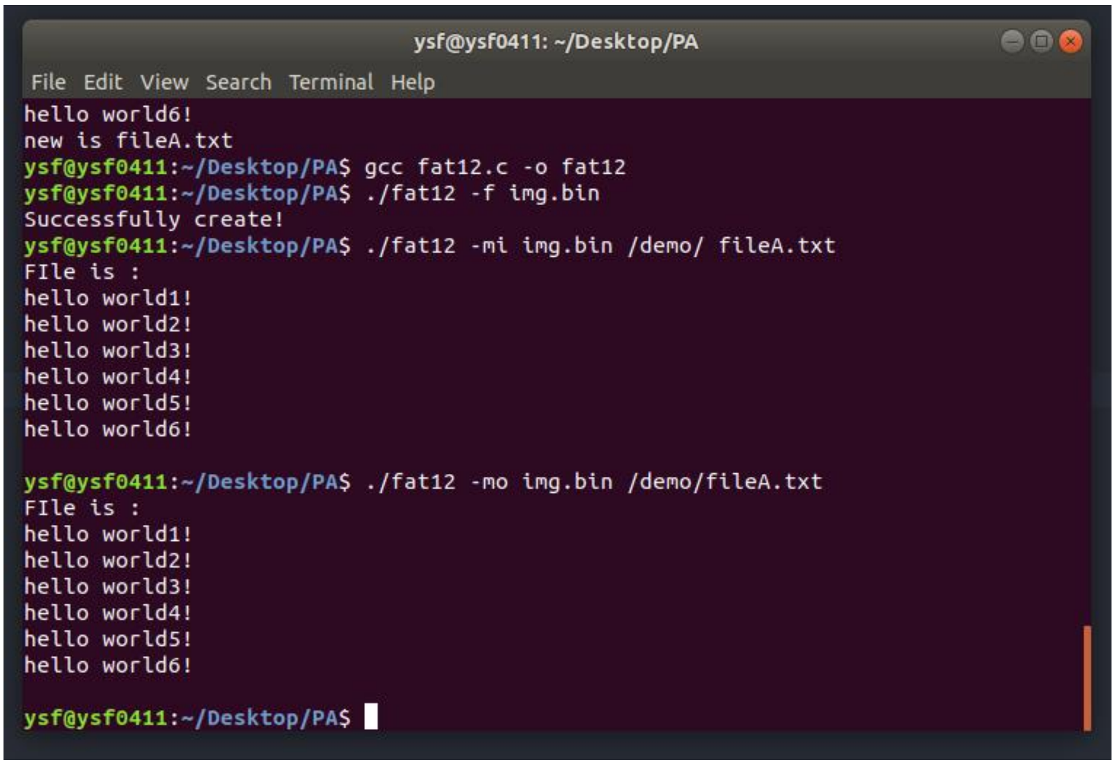
为了让实验效果看起来更好，我在每次文件复制结束之后把文件内容进行了打印.



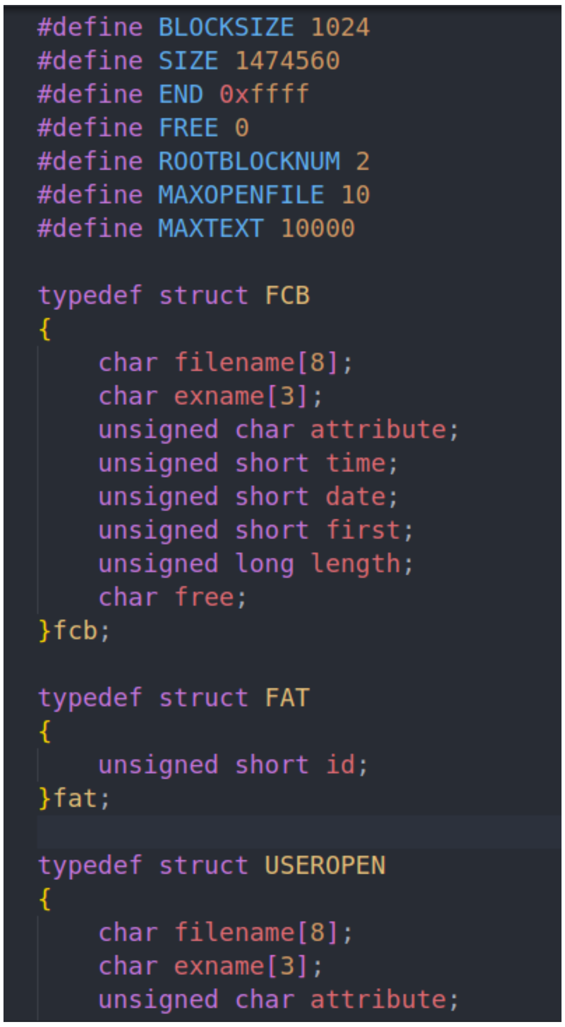
```
hello world1!
hello world2!
hello world3!
hello world4!
hello world5!
hello world6!
```

(这是原文件的内容，最后一行有回车)

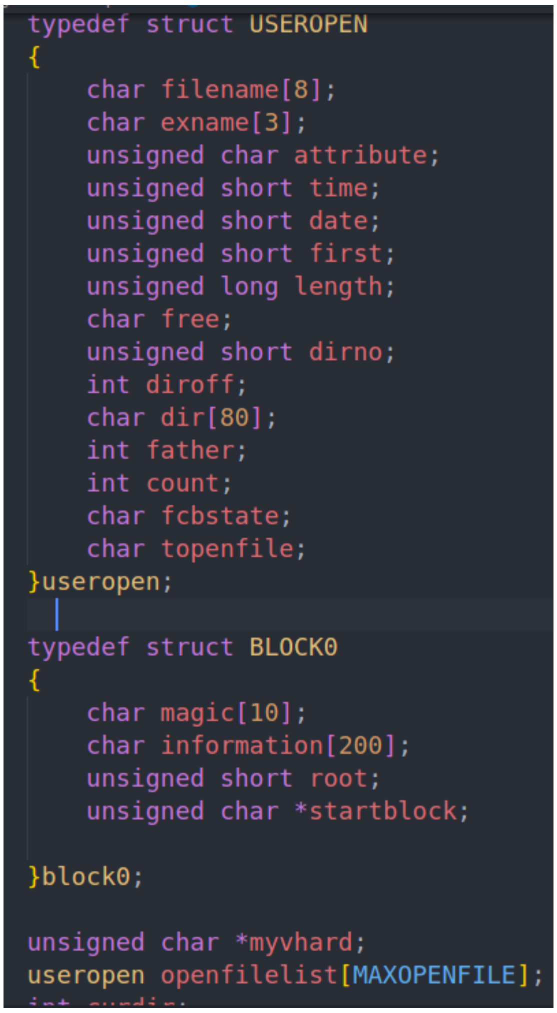
在这个编程作业中，我首先阅读了 FAT12 的框架代码，然后发现 FAT12 的簇定义需要 12 个字节，实现太过于麻烦，因此我使用了 16 个字节，这样会导致数据区变小，但是这样的速度却比 12 个字节块，然后仿照 FAT12 的文件表，超级块定义了一系列的结构体以供使用.


ysf@ysf0411: ~/Desktop/PA
File Edit View Search Terminal Help

```
hello world6!
new is fileA.txt
ysf@ysf0411:~/Desktop/PA$ gcc fat12.c -o fat12
ysf@ysf0411:~/Desktop/PA$ ./fat12 -f img.bin
Successfully create!
ysf@ysf0411:~/Desktop/PA$ ./fat12 -mi img.bin /demo/ fileA.txt
FIle is :
hello world1!
hello world2!
hello world3!
hello world4!
hello world5!
hello world6!

ysf@ysf0411:~/Desktop/PA$ ./fat12 -mo img.bin /demo/fileA.txt
FIle is :
hello world1!
hello world2!
hello world3!
hello world4!
hello world5!
hello world6!

ysf@ysf0411:~/Desktop/PA$
```



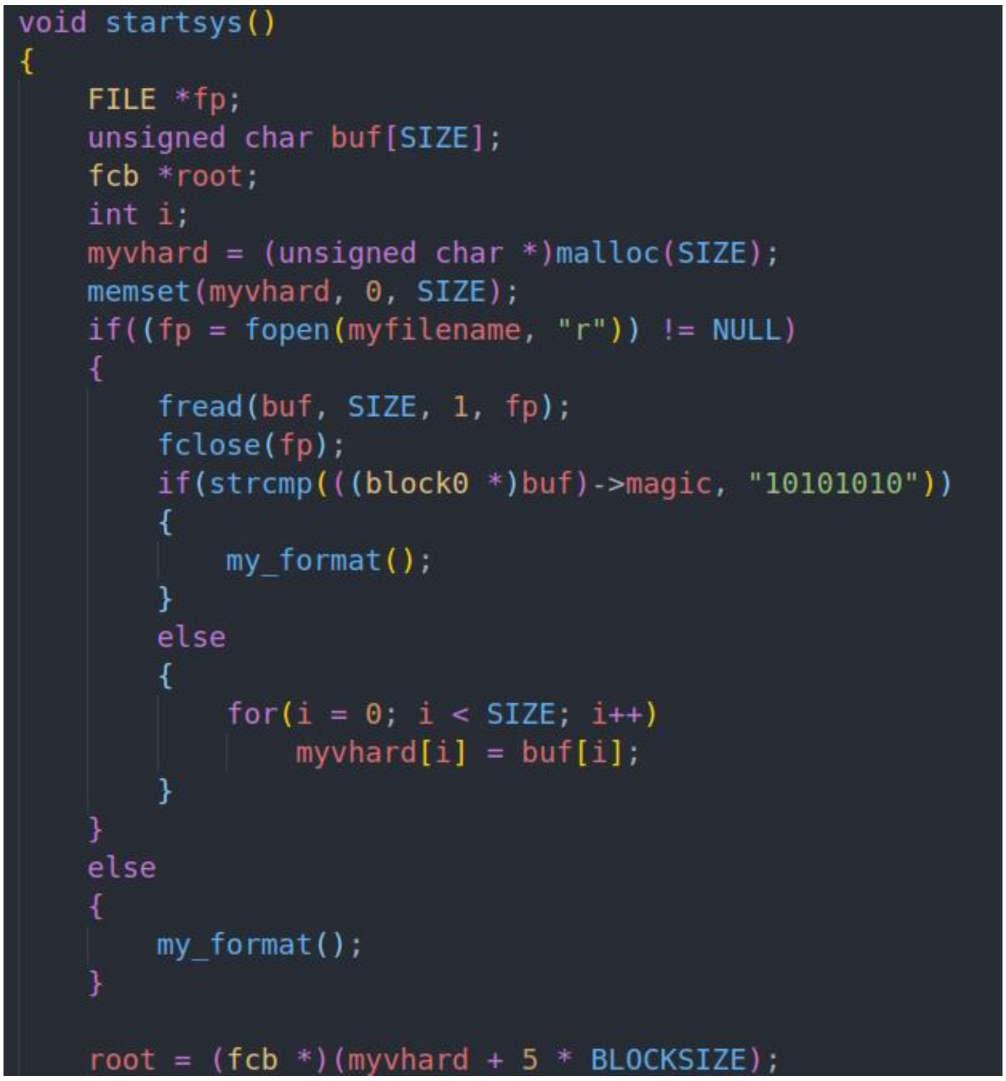
```c
#define BLOCKSIZE 1024
#define SIZE 1474560
#define END 0xffff
#define FREE 0
#define ROOTBLOCKNUM 2
#define MAXOPENFILE 10
#define MAXTEXT 10000

typedef struct FCB
{
    char filename[8];
    char exname[3];
    unsigned char attribute;
    unsigned short time;
    unsigned short date;
    unsigned short first;
    unsigned long length;
    char free;
}fcb;

typedef struct FAT
{
    unsigned short id;
}fat;

typedef struct USEROPEN
{
    char filename[8];
    char exname[3];
    unsigned char attribute;
```



```c
typedef struct USEROPEN
{
    char filename[8];
    char exname[3];
    unsigned char attribute;
    unsigned short time;
    unsigned short date;
    unsigned short first;
    unsigned long length;
    char free;
    unsigned short dirno;
    int diroff;
    char dir[80];
    int father;
    int count;
    char fcbstate;
    char topenfile;
}useropen;

typedef struct BLOCK0
{
    char magic[10];
    char information[200];
    unsigned short root;
    unsigned char *startblock;

}block0;

unsigned char *myvhard;
useropen openfilelist[MAXOPENFILE];
```

然后使用这些数据结构进行 FAT12 的模拟.

首先是创建:



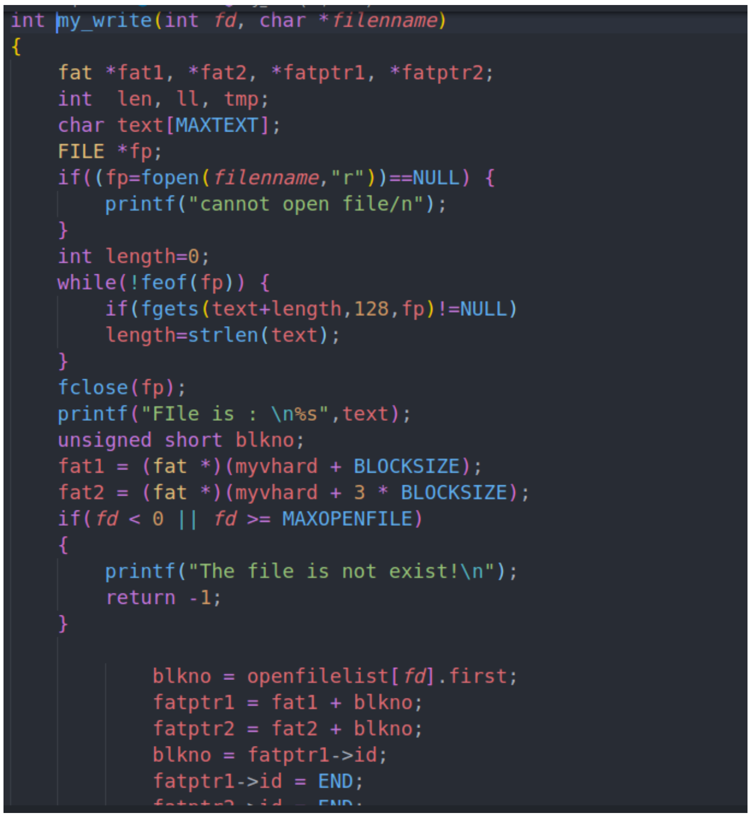
```c
void startsys()
{
    FILE *fp;
    unsigned char buf[SIZE];
    fcb *root;
    int i;
    myvhard = (unsigned char *)malloc(SIZE);
    memset(myvhard, 0, SIZE);
    if((fp = fopen(myfilename, "r")) != NULL)
    {
        fread(buf, SIZE, 1, fp);
        fclose(fp);
        if(strcmp(((block0 *)buf)->magic, "10101010"))
        {
            my_format();
        }
        else
        {
            for(i = 0; i < SIZE; i++)
                myvhard[i] = buf[i];
        }
    }
    else
    {
        my_format();
    }

    root = (fcb *)(myvhard + 5 * BLOCKSIZE);
```

我会首先打开文件名的文件，如果存在，比较魔数是否相同，若不同，则需要格式化整个文件，如果相同，直接使用即可。

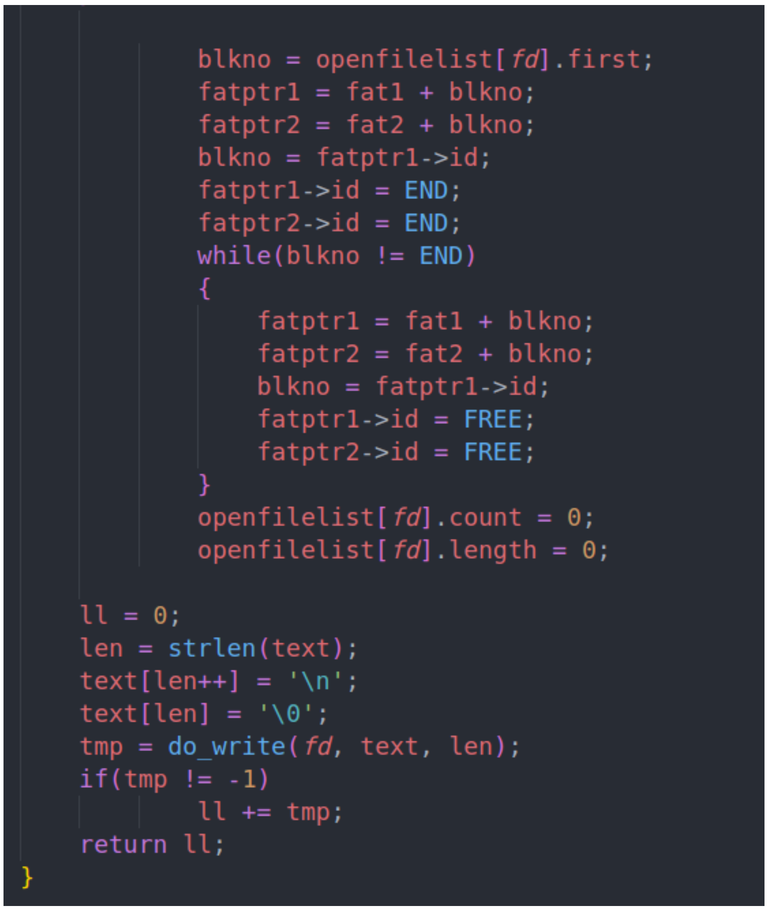
格式化文件，需要对文件的根目录和引导块进行设置，并且设置魔数以便于下次创建。

然后是复制进 FAT12:

我会首先打开这个文件，然后将主机上的文件内容拷贝进 text 数组中，然后将这个数组内的内容写到我的 FAT12 文件系统中，这里如果没有文件会直接报错，这里需要主义的地方是需要对块的属性进行设置，以便于文件的打开。



```c
int my_write(int fd, char *filename)
{
    fat *fat1, *fat2, *fatptr1, *fatptr2;
    int  len, ll, tmp;
    char text[MAXTEXT];
    FILE *fp;
    if((fp=fopen(filename,"r"))==NULL) {
        printf("cannot open file/n");
    }
    int length=0;
    while(!feof(fp)) {
        if(fgets(text+length,128,fp)!=NULL)
        length=strlen(text);
    }
    fclose(fp);
    printf("FIle is : \n%s",text);
    unsigned short blkno;
    fat1 = (fat *)(myvhard + BLOCKSIZE);
    fat2 = (fat *)(myvhard + 3 * BLOCKSIZE);
    if(fd < 0 || fd >= MAXOPENFILE)
    {
        printf("The file is not exist!\n");
        return -1;
    }

            blkno = openfilelist[fd].first;
            fatptr1 = fat1 + blkno;
            fatptr2 = fat2 + blkno;
            blkno = fatptr1->id;
            fatptr1->id = END;
```

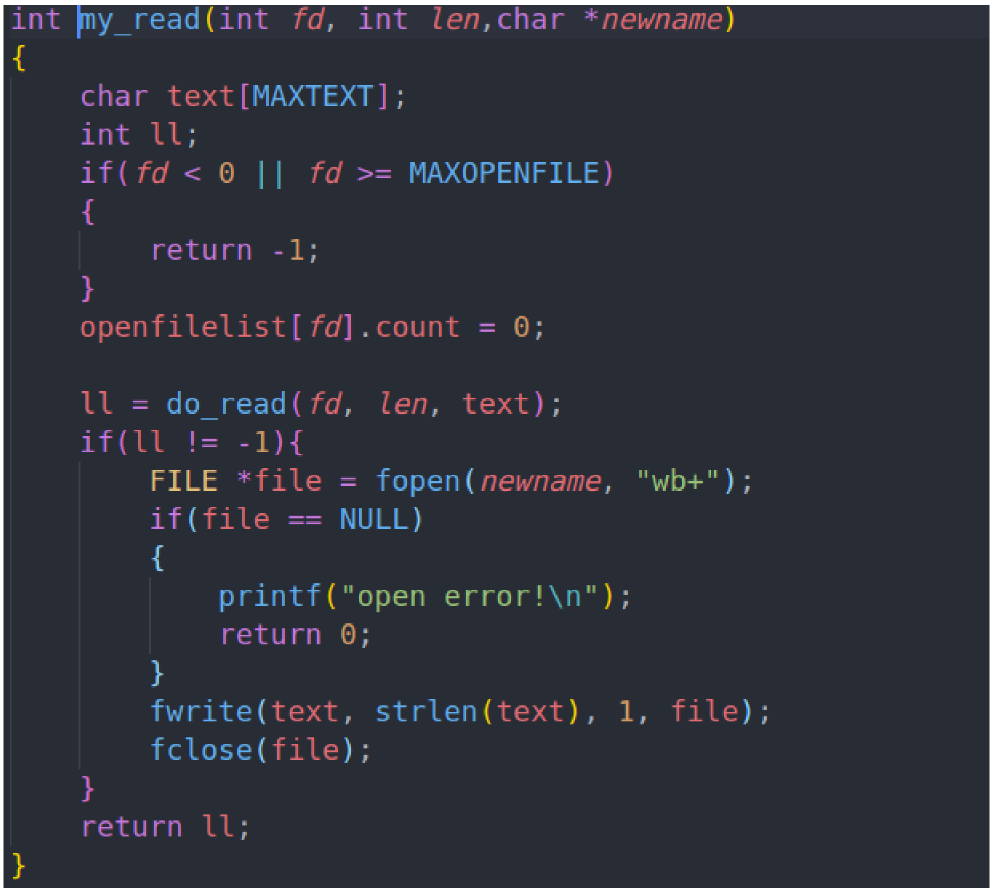


```c
            blkno = openfilelist[fd].first;
            fatptr1 = fat1 + blkno;
            fatptr2 = fat2 + blkno;
            blkno = fatptr1->id;
            fatptr1->id = END;
            fatptr2->id = END;
            while(blkno != END)
            {
                fatptr1 = fat1 + blkno;
                fatptr2 = fat2 + blkno;
                blkno = fatptr1->id;
                fatptr1->id = FREE;
                fatptr2->id = FREE;
            }
            openfilelist[fd].count = 0;
            openfilelist[fd].length = 0;

    ll = 0;
    len = strlen(text);
    text[len++] = '\n';
    text[len] = '\0';
    tmp = do_write(fd, text, len);
    if(tmp != -1)
            ll += tmp;
    return ll;
}
```

然后是文件的复制出系统：

我首先打开了文件，然后还是同样的把文件内容读取到 text 中，然后把 text 的内容

在主系统中写到新文件中即可。



```c
int my_read(int fd, int len,char *newname)
{
    char text[MAXTEXT];
    int ll;
    if(fd < 0 || fd >= MAXOPENFILE)
    {
        return -1;
    }
    openfilelist[fd].count = 0;

    ll = do_read(fd, len, text);
    if(ll != -1){
        FILE *file = fopen(newname, "wb+");
        if(file == NULL)
        {
            printf("open error!\n");
            return 0;
        }
        fwrite(text, strlen(text), 1, file);
        fclose(file);
    }
    return ll;
}
```