

# Operating System

## 2024, Spring

Homework | Project - 02

Ahmet Yigit - 200104004066

---

- **System Design & Important Questions Answers**

- **File System Structure**

- **SuperBlock:** Contains metadata about the file system.

```
1  typedef struct {  
2      int block_size;  
3      int total_blocks;  
4      int free_blocks;  
5      int fat_start;  
6      int entry_start;  
7      int total_entries;  
8      int directory_start;  
9      int total_directories;  
10     int data_start;  
11 } SuperBlock;  
12
```

- **FAT (File Allocation Table):** Keeps track of which blocks are used and which are free.
    - **Directory Entries:** Stores metadata for files, including size, creation time, permissions, etc.

```
1  typedef struct {  
2      int size;  
3      int dataSize;  
4      int startBlock;  
5      int entryOrder;  
6      char permissions[2];  
7      time_t creation_time;  
8      time_t modification_time;  
9      int isPasswordProtected;  
10     char parentDirectory[MAX_DIRECTORY_NAME];  
11     char password[32];  
12 } DirectoryEntry;
```

- **Directories:** Manages the hierarchy of directories within the file system. It behaves as a directory table.

```
1 typedef struct {
2     char name[MAX_DIRECTORY_NAME];
3     int size;
4     int startBlock;
5     time_t creation_time;
6 } Directory;
```

- 
- **Data Blocks:** Stores the actual file data.

- **Keep Tracking Free Blocks**

- System holds free blocks in an array. When a new entry comes it searches all disks and where is the not used which is defined as [-1,0,1] in the integer array, helps where is the empty block and locate it.

```
1 void findEmptyBlock
2     (FILE* fs, SuperBlock* sb, int* block_number) {
3
4     int* fat = (int*)malloc(sb->total_blocks * sizeof(int));
5     fseek(fs, sb->fat_start * sb->block_size, SEEK_SET);
6     fread(fat, sizeof(int), sb->total_blocks, fs);
7     for (int i = sb->data_start; i < sb->total_blocks; i++) {
8         if (fat[i] == -1) {
9             *block_number = i;
10            break;
11        }
12    }
13    free(fat);
14
15 }
```

- Int pointer assigned as first possible available block\_number.

- **Solving of Arbitrary length of file name problem**

- In normal system design they hold in directoryEntry struct in fixed maximum size, but we have to handle not known size so i hold filename as data because in data blocks we can hold any size of char array because it is possible to use as long as our memory is enough.
- My strategy is to write a file name and then add ">" and add data to the data block.
  - Like my filename is test.txt and contains ahmetyigitdatasi
    - test.txt>ahmetyigitdatasi
- I select ">" because in rule most OS file names can not contain ">" characters.

---

```

1 // add targetDirectory+>+data
2 for(int j = 0; j < strlen(targetDirectory); j++){
3     data[j] = targetDirectory[j];
4 }
5 data[strlen(targetDirectory)] = '>';
6
7 for(int j = 0; j < size; j++){
8     data[j + strlen(targetDirectory) + 1] = fgetc(source);
9 }
10
11 // .. add data to located block

```

- **Handling permissions and password protection**

- Handling permissions:

- I wrote a basic algorithm in the addpw function.
                - It checks given entry is found or not
                - It checks user entered minus or plus
                  - if minus it removes what user give
                  - If plus it add what user give
                    - simple character checking in char array.
                - After addition I save this information in DirectorEntry struct data and save it.

- Password protection

- It checks given entry is found or not
                - It controls the length of password 3-32.
                - It takes from the user and adds it to the directoryEntry struct.
                - When reading this entry if it is a passwordProtected system checks if the user gives a password and if it gives the password it matches.
                - After checking it can be read to block.

- **System Operation Functions**

- **IN CODE I DEFINED `ROOT_DEFAULT_PATH='/'` YOU CAN CHANGE TO USE `"\"` BUT NOT FORGOT `"\"` IS A ESCAPE CHARACTER SO YOU HAVE TO WRITE `\\` ROOT DIRECTORY.**
    - **All included screenshots applied after made of the given test cases in the homework pdf.**

- DIR

- `int dir(FILE *fs, SuperBlock *sb, const char* targetDirectory, int isPrint)`

The screenshot shows a VS Code editor with the following files open: `makeFileSystem.c`, `FileSystemOper.c`, `Makefile`, and `linuxFile.data`. The `Makefile` contains the following rules:

```

1 disk:
2     gcc -o makeFileSystem makeFileSystem.c
3     ./makeFileSystem 1 fileSystem.data
4 ops:
5     gcc -o fileSystemOper fileSystemOper.c
6
7 clean:
8     rm makeFileSystem fileSystemOper
9
10 ilhanHoca:
11     ./fileSystemOper fileSystem.data mkdir /usr
12     ./fileSystemOper fileSystem.data mkdir /usr/ysa
13     ./fileSystemOper fileSystem.data mkdir /bin/ysa
14     ./fileSystemOper fileSystem.data write /usr/ysa/file1 ./linuxFile.data
15     ./fileSystemOper fileSystem.data write /usr/file2 ./linuxFile.data
16     ./fileSystemOper fileSystem.data write /file3 ./linuxFile.data
17     ./fileSystemOper fileSystem.data dir /
18     ./fileSystemOper fileSystem.data del /usr/ysa/file1
19     ./fileSystemOper fileSystem.data dmp2fs
20     ./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
21     cmp ./linuxFile.data ./linuxFile2.data
22     ./fileSystemOper fileSystem.data chmod /usr/file2 -rw
23     ./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
24     ./fileSystemOper fileSystem.data chmod /usr/file2 +rw
25     ./fileSystemOper fileSystem.data addpw /usr/file2 test1234
26     ./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
27     ./fileSystemOper fileSystem.data read /usr/file2 ./linuxFilePassword.data test1234

```

The terminal window shows the following commands and output:

```

mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ make disk
gcc -o makeFileSystem makeFileSystem.c
./makeFileSystem 1 fileSystem.data
mens1s@mens1s:~/Downloads/osTest$ make ops
gcc -o fileSystemOper fileSystemOper.c
mens1s@mens1s:~/Downloads/osTest$ make ilhanHoca

```

The terminal window shows the output of the `dir` function:

```

mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /file3
Size: 0
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024

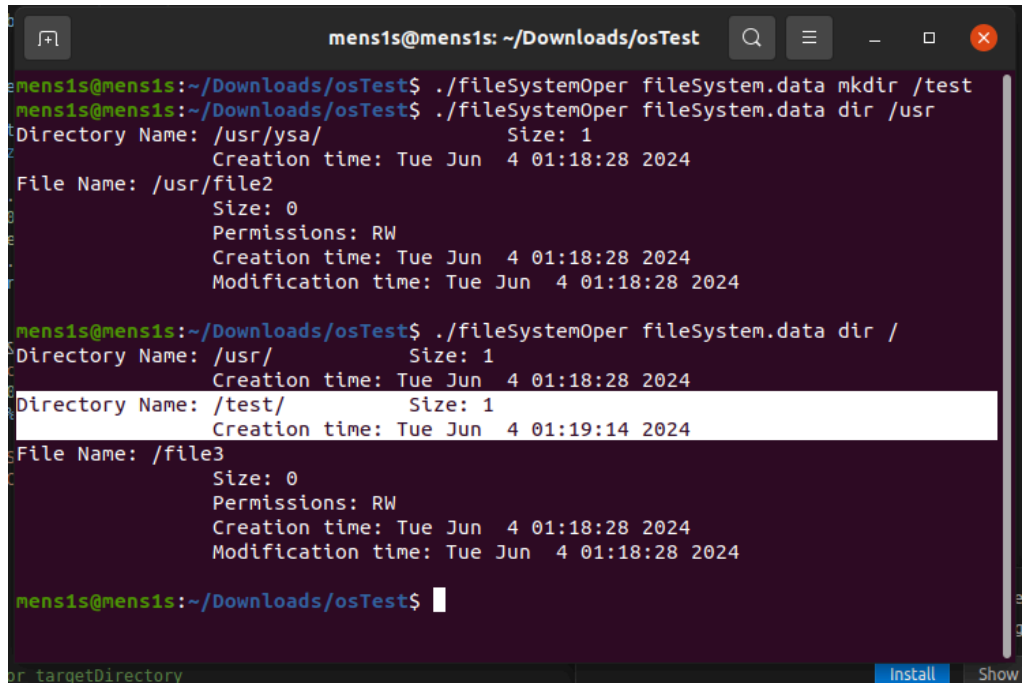
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /usr
Directory Name: /usr/ysa/      Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /usr/file2
Size: 0
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024

mens1s@mens1s:~/Downloads/osTest$

```

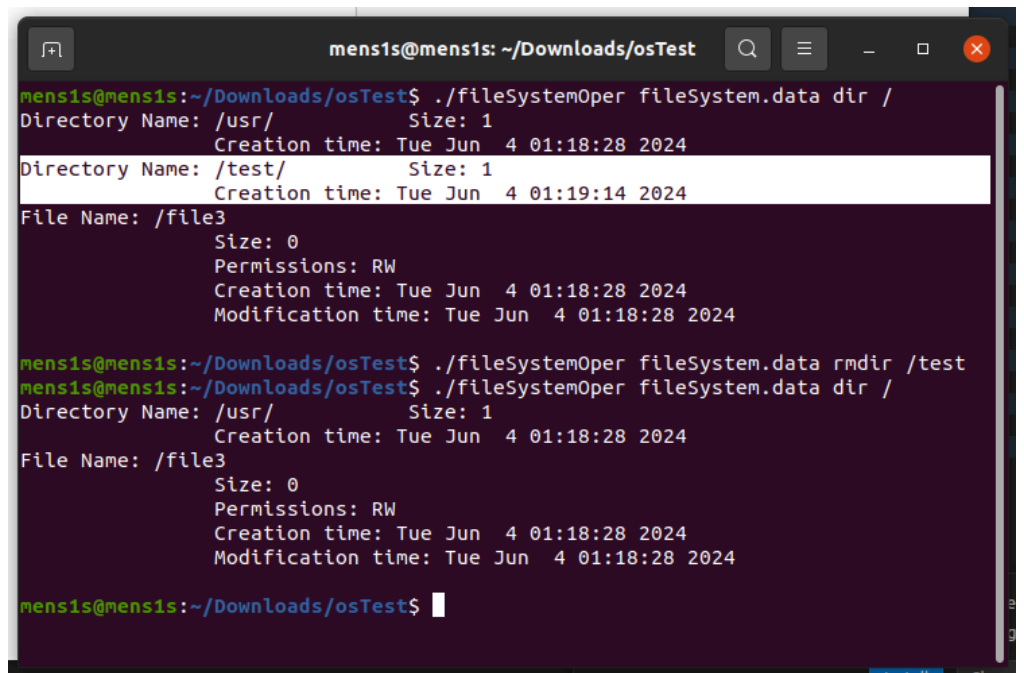
- o mkdir

- void mkdir(FILE \*fs, SuperBlock \*sb, const char\* targetDirectory)
  - Second K is upperCase because it conflict with another function name which comes from library.

A terminal window titled 'mens1s@mens1s: ~/Downloads/osTest' showing the execution of a file system operation. The user runs './fileSystemOper fileSystem.data mkdir /test'. The output shows the creation of a directory '/usr/ysa/' with size 1, creation time Tue Jun 4 01:18:28 2024, and a file '/usr/file2' with size 0, permissions RW, creation time Tue Jun 4 01:18:28 2024, and modification time Tue Jun 4 01:18:28 2024. The user then runs './fileSystemOper fileSystem.data dir /'. The output shows the creation of a directory '/usr/' with size 1, creation time Tue Jun 4 01:18:28 2024, and a directory '/test/' with size 1, creation time Tue Jun 4 01:19:14 2024. The user then runs './fileSystemOper fileSystem.data dir /'. The output shows the creation of a file '/file3' with size 0, permissions RW, creation time Tue Jun 4 01:18:28 2024, and modification time Tue Jun 4 01:18:28 2024. The terminal prompt is 'mens1s@mens1s:~/Downloads/osTest\$'.

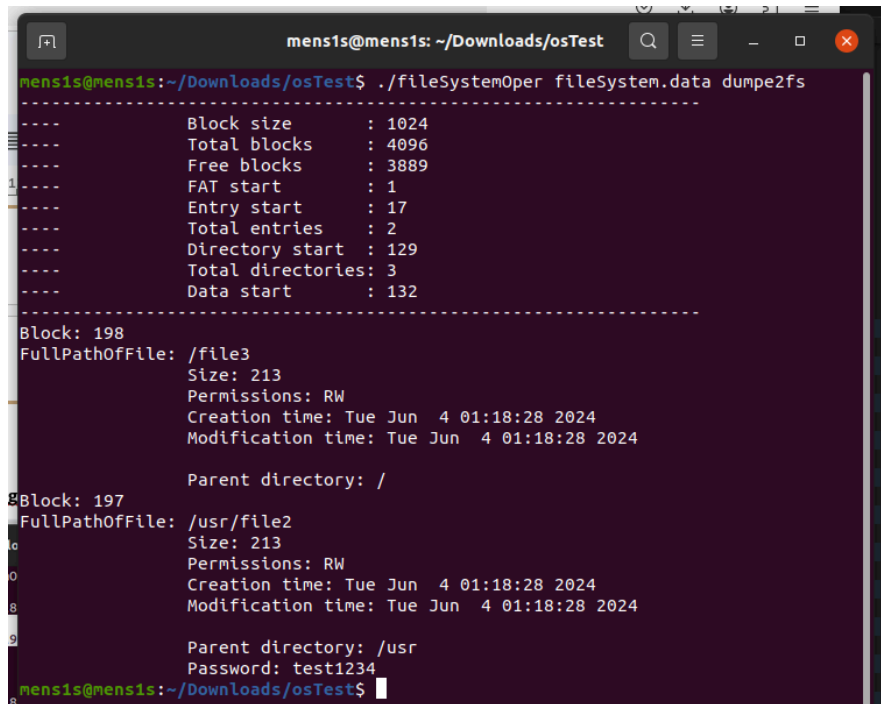
- o rmdir

- void rmdir(FILE \*fs, SuperBlock \*sb, const char\* targetDirectory)

A terminal window titled 'mens1s@mens1s: ~/Downloads/osTest' showing the execution of a file system operation. The user runs './fileSystemOper fileSystem.data dir /'. The output shows the creation of a directory '/usr/' with size 1, creation time Tue Jun 4 01:18:28 2024, and a directory '/test/' with size 1, creation time Tue Jun 4 01:19:14 2024. The user then runs './fileSystemOper fileSystem.data rmdir /test'. The output shows the removal of the directory '/test/'. The user then runs './fileSystemOper fileSystem.data dir /'. The output shows the creation of a directory '/usr/' with size 1, creation time Tue Jun 4 01:18:28 2024, and a file '/file3' with size 0, permissions RW, creation time Tue Jun 4 01:18:28 2024, and modification time Tue Jun 4 01:18:28 2024. The terminal prompt is 'mens1s@mens1s:~/Downloads/osTest\$'.

- o **dumpe2fs**

- **void dump2fs(SuperBlock \*sb, FILE \*fs)**

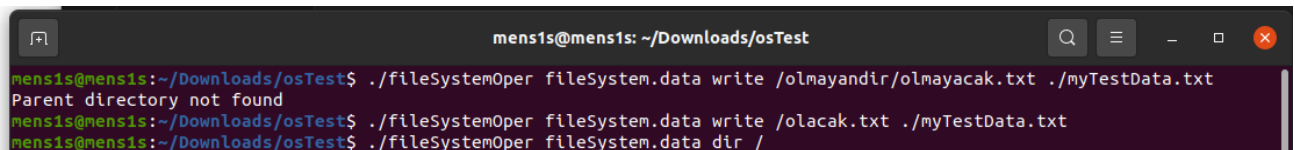


```
mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystem0per fileSystem.data dumpe2fs
-----
Block size      : 1024
Total blocks    : 4096
Free blocks     : 3889
FAT start       : 1
Entry start     : 17
Total entries   : 2
Directory start : 129
Total directories: 3
Data start      : 132
-----
Block: 198
FullPathOfFile: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
Parent directory: /
Block: 197
FullPathOfFile: /usr/file2
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
Parent directory: /usr
Password: test1234
mens1s@mens1s:~/Downloads/osTest$
```

- o **write**

- **void write(FILE \*fs, SuperBlock \*sb, const char\* sourceDirectory, const char\* targetDirectory)**

- **FIRST WRITE**



```
mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystem0per fileSystem.data write /olmayandır/olmayacak.txt ./myTestData.txt
Parent directory not found
mens1s@mens1s:~/Downloads/osTest$ ./fileSystem0per fileSystem.data write /olacak.txt ./myTestData.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystem0per fileSystem.data dir /
```

## ■ UPDATE

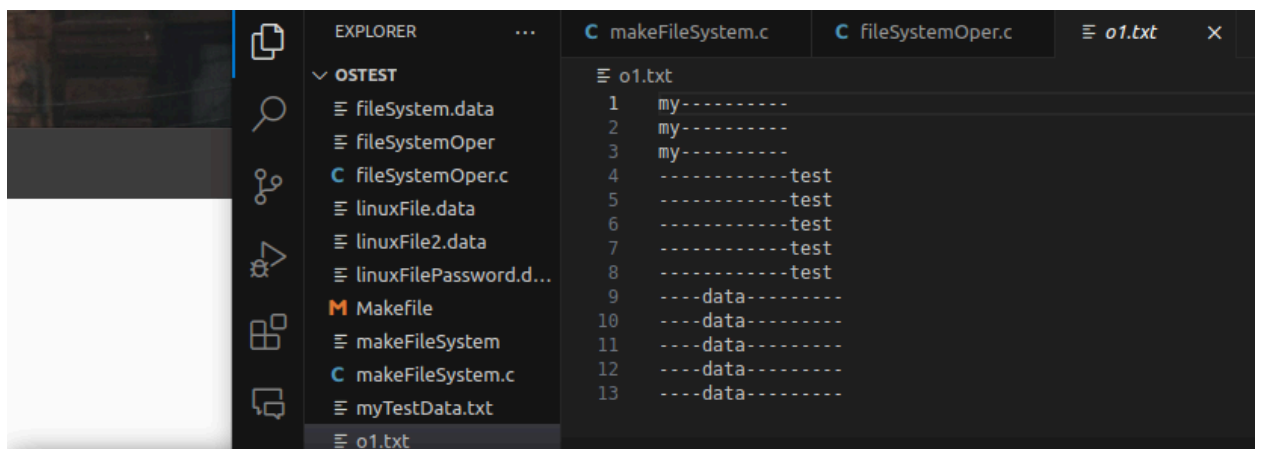
```

mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
File Name: /olacak.txt
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:23:37 2024
Modification time: Tue Jun  4 01:23:37 2024
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data write /olacak.txt ./myTestData.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
File Name: /olacak.txt
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:23:37 2024
Modification time: Tue Jun  4 01:26:19 2024
mens1s@mens1s:~/Downloads/osTest$

```

### ○ read

- `void read(FILE *fs, SuperBlock *sb, const char* sourceDirectory, const char* targetDirectory, const char* password)`



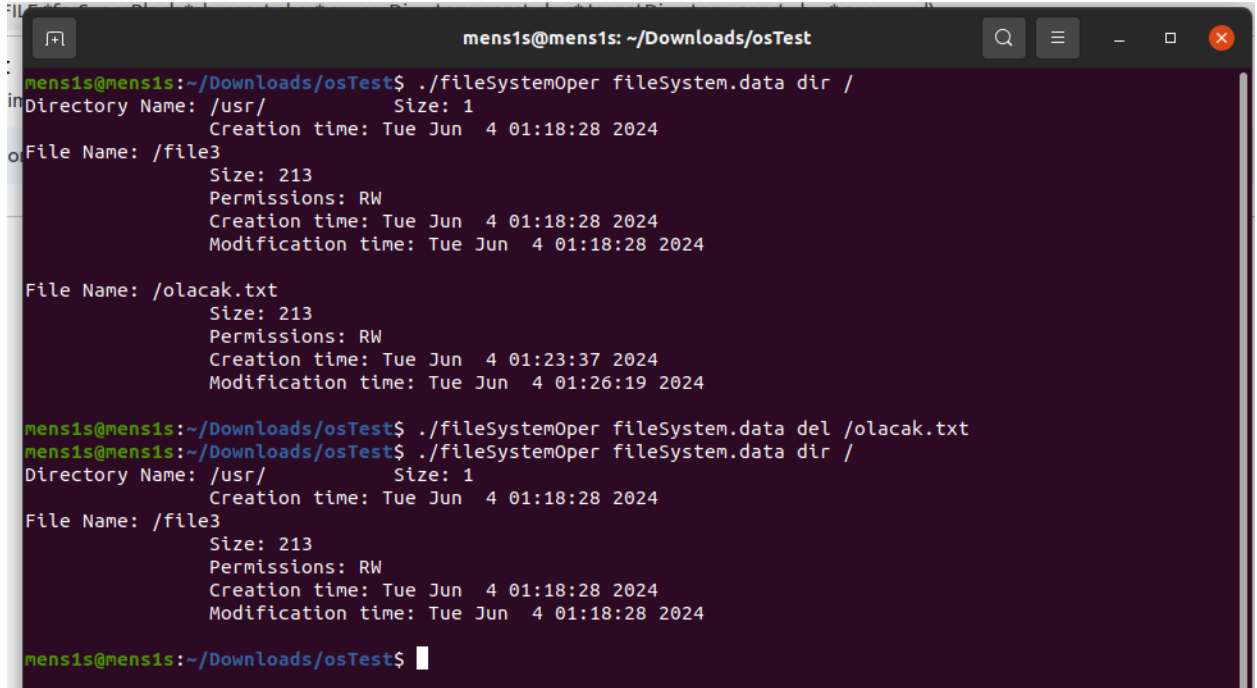
```

mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data read /olacak.txt ./o1.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data read /olacak.txt ./o1.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data read /olmayan.txt ./o1.txt
Source directory not found
mens1s@mens1s:~/Downloads/osTest$

```

- del

- void del(FILE \*fs, SuperBlock \*sb, const char \*targetDirectory)



```
mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
File Name: /olacak.txt
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:23:37 2024
Modification time: Tue Jun  4 01:26:19 2024
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data del /olacak.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:18:28 2024
File Name: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:18:28 2024
Modification time: Tue Jun  4 01:18:28 2024
mens1s@mens1s:~/Downloads/osTest$
```



- **chmod**

- **void chmod(FILE \*fs, SuperBlock \*sb, const char \*targetDirectory, char \*permissions)**

- Second H is upperCase because it conflict with another function name which comes from library.

```
mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:31:18 2024
File Name: /file3
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:32:03 2024

mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data chmod /file3 -rw
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dir /
Directory Name: /usr/          Size: 1
Creation time: Tue Jun  4 01:31:18 2024
File Name: /file3
Size: 213
Permissions: --
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:32:15 2024

mens1s@mens1s:~/Downloads/osTest$ ls
fileSystem.data  fileSystemOper.c  linuxFile.data      Makefile      makeFileSystem.c  o1.txt
fileSystemOper  linuxFile2.data   linuxFilePassword.data  makeFileSystem  myTestData.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data read /file3 ./olusturulmayacak.txt
No permission to read
mens1s@mens1s:~/Downloads/osTest$ ls
fileSystem.data  fileSystemOper.c  linuxFile.data      Makefile      makeFileSystem.c  o1.txt
fileSystemOper  linuxFile2.data   linuxFilePassword.data  makeFileSystem  myTestData.txt
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data write /file3 ./
fileSystem.data  linuxFile2.data   Makefile      myTestData.txt
fileSystemOper  linuxFile.data    makeFileSystem  o1.txt
fileSystemOper.c  linuxFilePassword.data  makeFileSystem.c
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data write /file3 ./myTestData.txt
No permission to write
mens1s@mens1s:~/Downloads/osTest$
```

- addpw

- void addPw(FILE \*fs, SuperBlock \*sb, const char \*targetDirectory, const char \*password)

```
-----
Block: 198
FullPathOfFile: /file3
Size: 213
Permissions: --
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:32:15 2024

Parent directory: /

Block: 197
FullPathOfFile: /usr/file2
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:31:18 2024

Parent directory: /usr
Password: test1234
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data addpw /file3 notpassword
mens1s@mens1s:~/Downloads/osTest$ ./fileSystemOper fileSystem.data dume2fs
-----
----          Block size      : 1024
----          Total blocks    : 4096
----          Free blocks     : 3890
----          FAT start       : 1
----          Entry start      : 17
----          Total entries    : 2
----          Directory start  : 129
----          Total directories: 3
----          Data start       : 132
-----

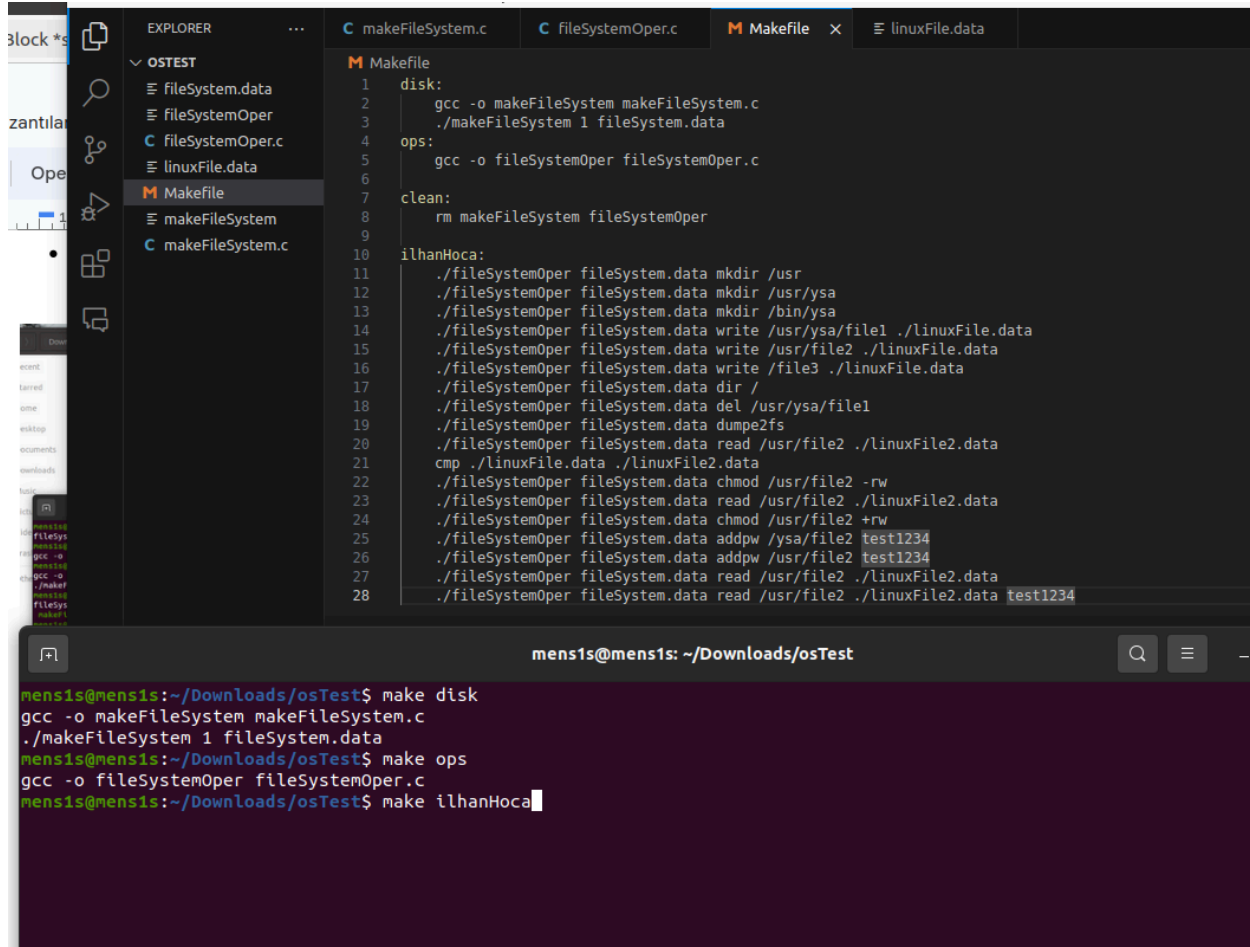
Block: 198
FullPathOfFile: /file3
Size: 213
Permissions: --
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:32:15 2024

Parent directory: /
Password: notpassword

Block: 197
FullPathOfFile: /usr/file2
Size: 213
Permissions: RW
Creation time: Tue Jun  4 01:31:18 2024
Modification time: Tue Jun  4 01:31:18 2024

Parent directory: /usr
Password: test1234
mens1s@mens1s:~/Downloads/osTest$
```

- TEST CASES
  - Homework Test Case
- Not Run



The image shows a VS Code editor with a project named 'osTest'. The Explorer sidebar on the left shows the file structure: `FileSystem.data`, `FileSystemOper`, `FileSystemOper.c`, `linuxFile.data`, `Makefile`, `makeFileSystem`, and `makeFileSystem.c`. The Makefile editor shows the following content:

```
1 disk:
2     gcc -o makeFileSystem makeFileSystem.c
3     ./makeFileSystem 1 FileSystem.data
4 ops:
5     gcc -o FileSystemOper FileSystemOper.c
6
7 clean:
8     rm makeFileSystem FileSystemOper
9
10 ilhanHoca:
11     ./FileSystemOper FileSystem.data mkdir /usr
12     ./FileSystemOper FileSystem.data mkdir /usr/ysa
13     ./FileSystemOper FileSystem.data mkdir /bin/ysa
14     ./FileSystemOper FileSystem.data write /usr/ysa/file1 ./linuxFile.data
15     ./FileSystemOper FileSystem.data write /usr/file2 ./linuxFile.data
16     ./FileSystemOper FileSystem.data write /file3 ./linuxFile.data
17     ./FileSystemOper FileSystem.data dir /
18     ./FileSystemOper FileSystem.data del /usr/ysa/file1
19     ./FileSystemOper FileSystem.data dume2fs
20     ./FileSystemOper FileSystem.data read /usr/file2 ./linuxFile2.data
21     cmp ./linuxFile.data ./linuxFile2.data
22     ./FileSystemOper FileSystem.data chmod /usr/file2 -rw
23     ./FileSystemOper FileSystem.data read /usr/file2 ./linuxFile2.data
24     ./FileSystemOper FileSystem.data chmod /usr/file2 +rw
25     ./FileSystemOper FileSystem.data addpw /ysa/file2 test1234
26     ./FileSystemOper FileSystem.data addpw /usr/file2 test1234
27     ./FileSystemOper FileSystem.data read /usr/file2 ./linuxFile2.data
28     ./FileSystemOper FileSystem.data read /usr/file2 ./linuxFile2.data test1234
```

Below the editor is a terminal window with the following commands and output:

```
mens1s@mens1s: ~/Downloads/osTest
mens1s@mens1s:~/Downloads/osTest$ make disk
gcc -o makeFileSystem makeFileSystem.c
./makeFileSystem 1 FileSystem.data
mens1s@mens1s:~/Downloads/osTest$ make ops
gcc -o FileSystemOper FileSystemOper.c
mens1s@mens1s:~/Downloads/osTest$ make ilhanHoca
```

## Result:

```
./fileSystemOper fileSystem.data mkdir /usr
./fileSystemOper fileSystem.data mkdir /usr/ysa
./fileSystemOper fileSystem.data mkdir /bin/ysa
Parent directory not found
./fileSystemOper fileSystem.data write /usr/ysa/file1 ./linuxFile.data
./fileSystemOper fileSystem.data write /usr/file2 ./linuxFile.data
./fileSystemOper fileSystem.data write /file3 ./linuxFile.data
./fileSystemOper fileSystem.data dir /
Directory Name: /usr/      Size: 1
      Creation time: Tue Jun  4 01:58:51 2024
File Name: /file3
      Size: 65
      Permissions: RW
      Creation time: Tue Jun  4 01:58:51 2024
      Modification time: Tue Jun  4 01:58:51 2024

./fileSystemOper fileSystem.data del /usr/ysa/file1
./fileSystemOper fileSystem.data dumpe2fs
-----
----      Block size      : 1024
----      Total blocks    : 4096
----      Free blocks     : 3891
----      FAT start       : 1
----      Entry start     : 17
----      Total entries    : 2
----      Directory start  : 129
----      Total directories: 3
----      Data start      : 132
-----
Block: 198
FullPathOfFile: /file3
      Size: 65
      Permissions: RW
      Creation time: Tue Jun  4 01:58:51 2024
      Modification time: Tue Jun  4 01:58:51 2024

      Parent directory: /

Block: 197
FullPathOfFile: /usr/file2
      Size: 65
      Permissions: RW
      Creation time: Tue Jun  4 01:58:51 2024
      Modification time: Tue Jun  4 01:58:51 2024

      Parent directory: /usr
./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
cmp ./linuxFile.data ./linuxFile2.data
./fileSystemOper fileSystem.data chmod /usr/file2 -rw
./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
No permission to read
./fileSystemOper fileSystem.data chmod /usr/file2 +rw
./fileSystemOper fileSystem.data addpw /ysa/file2 test1234
Entry not found
./fileSystemOper fileSystem.data addpw /usr/file2 test1234
./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data
Wrong password
./fileSystemOper fileSystem.data read /usr/file2 ./linuxFile2.data test1234
mens1s@mens1s:~/Downloads/osTest$
```

- 
- t