Gebze Technical University

Department of Computer Engineering

CSE 312 /CSE 504

Operating Systems

Spring 2020

Midterm Exam Project

REPORT

Student: **Nevra Gürses**

No: **161044071**

Instructor: **Yusuf Sinan Akgül**

Nevra Gürses
161044071

MY FILE SYSTEM DESIGN:

Before I define all parts of my file system, I will briefly explain my file system design. My file system layout simply like Figure 4-9. of our book that is:

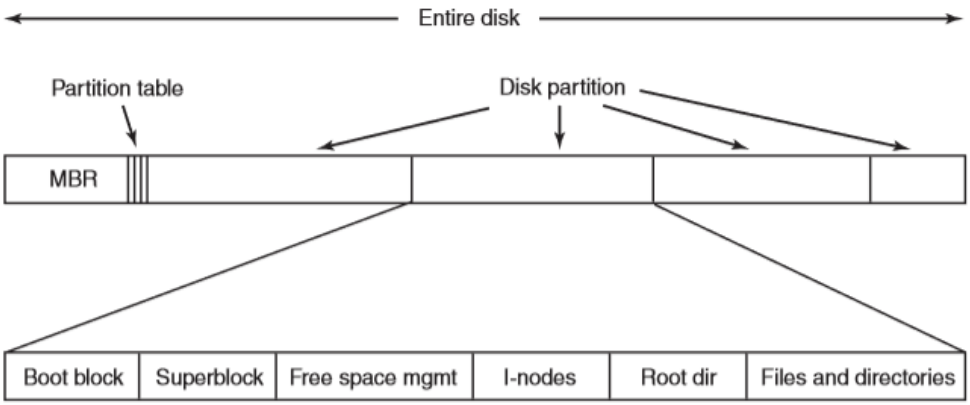


Figure 4-9. A possible file-system layout.

In my file system, disk is divided 5 part that are Superblock, Free Space Management, I-nodes, Root Directory, Files and Directories. In my file system, allocated area for Superblock part is **1 block size** that is given in commandline argument. Allocated area for Free Space Management part is also **1 block size**. Allocated area for I-nodes part is **%15 of block number** that is calculated as (total disk size/ block size). Allocated area for root directory part is **1 block size** and allocated area for files and directories part is **4 block size.** Remaining space of the disk is allocated for store data blocks. So,data block number is also changing according to i-nodes number,block size and other factors.Now,I will define all parts of file system layout in my file system design.

SUPERBLOCK

In my file system,allocated area for superblock part is **1 block size** that is given in commandline argument.Superblock contains crucial information about the file system.It includes magic number of file,size of file,block number in file system,block size of file system,start address of free space management part,i-node block number,free i-nodes number,start address of i-node blocks,start address of root directory,number of directories in root directory,start adress of files and directories part,number of directories,number of files,data block number and start address of data blocks.To show this in my file system:

```
~~SUPERBLOCK~~
Magic number:0x12345
Size of file:1 MB
Block number:256
Block size:4096
Start address of free space management:4096
I-node block number:38
Free i-nodes number:400
Start address of i-node blocks:8192
Start address of root directory:163840
Number of directories in root dir:36
Start address of files and directories:167936
Number of directories:35
Number of Files:400
Data block number:211
Start address of data blocks:184320
```

Nevra Gürses
161044071

Now, I will explain means of these crucial informations about the file system.

- **Magic Number:**Actually magic number is used to identify the file-system type.I write magic number that is 0x12345 as symbolic.
- **Size of File:** That is total size of file system.In our project this size is 1 MB all the time whether it contains any information or not.
- **Block Number:** This is total block number in file system.Block number is calculated as (total file size/ block size).
- **Block Size:** Block size is size of block for both data blocks and i-node blocks. This size is given in commandline argument.As an example of our file system, this number 4 kb that is equal 4096 byte.
- **Start address of free space management:** This is start address of free space management part.This is calculated as 1 block size in my file system because superblock allocates 1 block size and after free space management part starts.
- **I-node block number:** This is total i-node block number that is calculated as (block number*15)/100 in my file system.Allocated area for i-node blocks in my file system is %15 of total block number.
- **Free i-nodes number :** This is number of free i-nodes in start of file system.This number is given in commandline argument.In my example,free i-nodes number is 400.
- **Start address of i-nodes block:** This is start adress of i-nodes blocks.This address is calculated as (2*fsmStart) in my file sysrem because fsm part and superblock part are allocated as 1 block size area and after i-nodes part is starting.
- **Start address of root directory:** This is start adress of root directory part.This address is calculated as (inodeStart + (inodeNum*blockSize)) because root directory part starts after i-node parts.
- **Number of directories in root directory:** This is total number of directories that root directory can keep.This number is calculated as (block size /113) in my file system because allocated area for root directory is 1 block size and one entry is allocating 113 byte.
- **Start address of files and directories:** This is start address of files and directories.This address is calculated as (rootDirStart + blockSize) because this part is starting after root directory part and root directory part is 1 block size.
- **Number of directories:** This is total number of directories .This number is calculated as (4*block size /458) in my file system because allocated area for files and directories are 4 block size and one entry is allocating 458 byte.
- **Number of files:** This is total number of files in file system.This number is equal to free i-node number because 1 i-node can keep 1 file.So,in my example this number is 400.
- **Data block number:** This is total data block number that is calculated as (blockNum -(inodeNum+7)).Total data block number is remaining space of file.
- **Start address of data blocks:** This is start address of data blocks.This is calculated as (fileDirStart+(4*blockSize)) in my file system because allocated area for files and directories is 4 block size and data blocks starts after files and directories part.

Nevra Gürses
161044071

FREE SPACE MANAGEMENT

In my file system,allocated area for fsm part is **1 block size.** This part is using to keep free spaces of file system.After every change in file system such as mkdir operation,write operation and etc this area is updating.In my file system that part is:

```
18  ~~FREE SPACE MGMT~~
19  Free i-nodes number:400
20  Free data block number:211
21  Free data block address:184320
22  Free root dir index:1
23  Free root file index:1
24  Free dir index:1
25  Free i-node index:1
```

Means of this informations:

- **Free i-nodes number:** Keeps free i-nodes in file system.After every write operation,this number is decreasing.
- **Free data block number:**Keeps free data blocks in file system.After every write operation free data block ,this number is decreasing according to file size. For example; if file size is 2 block size,free data block number is decreasing by 2.
- **Free data block adress:** Keeps first free data block adress.In start,this address is start address of data block section.After operations,this address is updating.
- **Free root dir index:** Keeps first free index of directory in root directory.In start,first directory is free.After operations,this index is updating.
- **Free root file index:** Keeps first free index of directory in root directory. In start,first file is free,after operations,this index is updating.
- **Free dir index:** Keeps first free index of directory in files and directories part.In start,first directory is free.After operations,this index is updating.
- **Free i-node index:** Keeps first free index of i-nodes. In start,first i-node is free,after operations,this index is updating.

I-NODE STRUCTURE

My i-node structure is like as Fig 4.33 of our textbook that is**:**

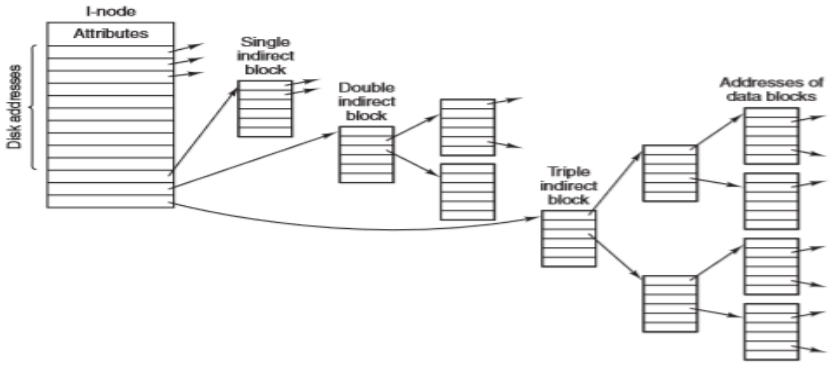


Figure 4-33. A UNIX i-node.

In my file system, allocated area for I-nodes part is **%15 of block number** that is calculated as (total disk size/ block size).One i-node block is 4 kb of our example and free i-nodes number is given as commandline argument.In our example, 400 is the number of free i-nodes for an empty file system.In my file system,i-node structure is:

```
~~I-NODES~~
I-node:1
Name of file:
Last modification date and time:
Size of file:
Disk-0 Address:
Disk-1 Address:
Disk-2 Address:
Disk-3 Address:
Disk-4 Address:
Disk-5 Address:
Single Indirect Block:
Double Indirect Block:
Triple Indirect Block:
```

(This is one i-node example,it goes this way.)

In my i-node structure;

- First line is index of i-node,in this example, index number is 1.
- After 3 lines are attributes of a file that are name of file, last modification date and time and size of file.
- Another 6 lines are disk addresses of file.
- Last 3 line are pointing indirect blocks.

I allocated 30 byte for name of file and creation time. For size of file,disk addresses and indirect blocks,I allocated 7 byte.

DIRECTORY STRUCTURE AND DIRECTORY ENTRIES

In my file system,there are 2 part for directories.

First; Root directory part keeps directories and files in root.Allocated area for root dir partiation is **1 block size.**

Second;Directories and files are kept in files and directories partiation.Allocated area for files and directories part is **4 block size.**

My directory, i-node block, data block structure will be similar to Fig 4.34 of the textbook that is:

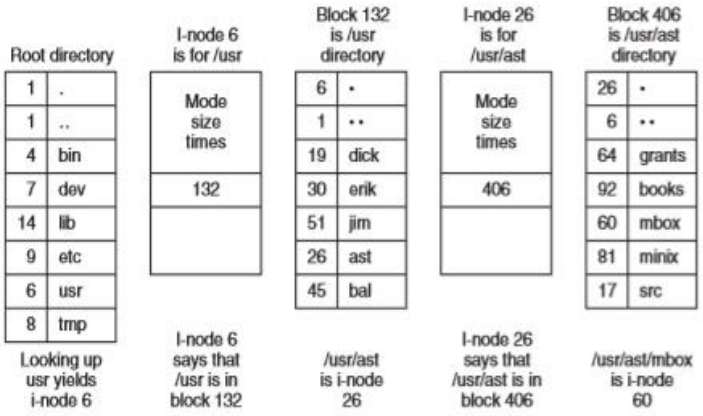


Figure 4-34. The steps in looking up */usr/ast/mbox*.

2.1 ROOT DIRECTORY PART

Keeps files and directories in root.That is shown in my file system like as:

```
~~ROOT DIR~~
R-Dir-1 Name:
R-File-1 Name:
R-I-node:
R-Dir-2 Name:
R-File-2 Name:
R-I-node:
R-Dir-3 Name:
R-File-3 Name:
R-I-node:
R-Dir-4 Name:
R-File-4 Name:
R-I-node:
```

(Goes in this way until last directory and file in root)

R-Dir-Index Name: Keeps directory name in root with index number.

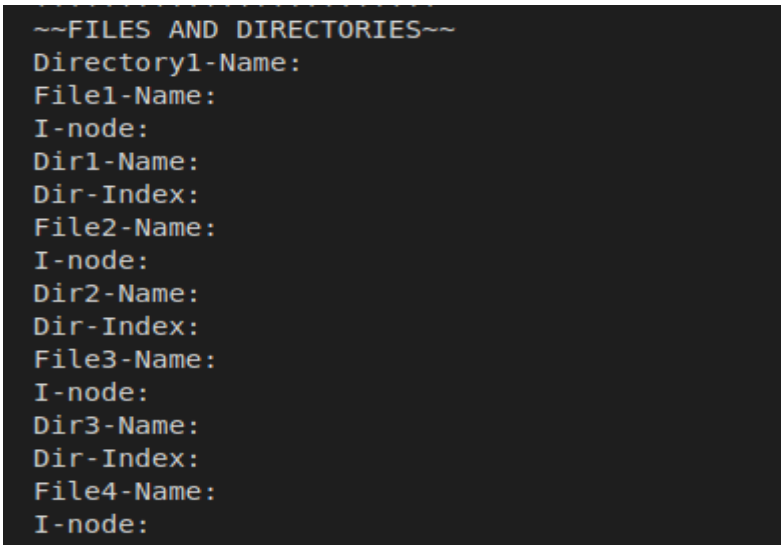
R-File-Index Name: Keeps file name in root with index number.

R-I-node: Keeps I-node number of one above file name.

NOTE: I allocated 30 byte for diretory and file names and I allocated 7 byte for to keep i-node indexes.

2.2 FILES AND DIRECTORIES PART

Keeps directories with inner files and directories .That is shown in my file system like as:



(Goes in this way until last directory.)

- **DirectoryIndex-Name**: Directory name with index.
- **File1-Name:** First file name in directory that is DirectoryIndex.
- **I-node**: I-node number of File1.
- **Dir1-Name**: First directory name in directory that is DirectoryIndex.
- **Dir-Index:** Index of Dir1.
- **File2-Name:** Second file name in directory that is DirectoryIndex.
- **I-node**: I-node number of File2.
- **Dir2-Name:** Second directory name in directory that is DirectoryIndex.
- **Dir-Index:** Index of Dir2.
- **File3-Name:** Third file name in directory that is DirectoryIndex.
- **I-node:** I-node number of File3.
- **Dir3-Name:** Third directory name in directory that is DirectoryIndex.
- **Dir-Index:** Index of Dir3.
- **File4-Name:** Fourth file name in directory that is DirectoryIndex.
- **I-node:** I-node number of File4.

NOTE: I allocated 30 byte for diretory and file names and I allocated 7 byte for to keep i-node indexes and directory indexes.

NOTE2: According to my file system,there can store at most 4 file and 3 directory in a directory.

NOTE3:Remaining parts after files and directories section is data blocks to keep files.

Nevra Gürses
161044071

FUNCTION NAMES THAT ARE USED IN PART 3:

File System Operations Functions:

- **void mkdirComm(FILE *fd,char* path,int fsmStart,int numberOfDir,int numberOfRootDir);**
- **void writeComm(FILE *fd,char* path,char* fileName,int fsmStart,int blockSize,int numberOfDir,int numberOfRootDir,int numberOfInode);**
- **int readComm(FILE *fd,char* path,char* fileName,int numberOfDir,int numberOfRootDir);**
- **void listComm(FILE* fd,char* path,int numberOfDir,int numberOfRootDir);**
- **void dumpe2fsComm(FILE* fd);**

Functions that are used to help File System Operations Functions:

- **int findString(FILE *fd,char*string,int start,int end);**
- **char* getString(FILE *fd,int byte);**
- **void change(FILE *fd,int byte,int size,char* changeStr);**
- **int searchInDir(FILE* fd,int amount,char* dirName);**
- **int searchInFile(FILE* fd,int amount,char*rootFile);**
- **void controlSameNameFile(FILE* fd,char* fileName,int start);**
- **void controlSameNameDir(FILE* fd,char* dirName,int start);**
- **void findFirstEmpty(FILE* fd,char pathArr[][100],int i,int numberOfDir);**
- **void findFirstEmptyFile(FILE* fd,char pathArr[][100],int i,int val,int numberOfDir);**
- **void helperControlPath(FILE*fd,char* dirName,char* innerDirName,int numberOfDir);**
- **void isValidPath(FILE* fd,char arrPath[][100],int num,int numberOfDir,int numberOfRootDir);**
- **void createDir(FILE *fd,char *dir,int fsmStart,int numberOfDir);**
- **void rootFileNameControl(FILE *fd,int amount,char* rootFile);**
- **void rootDirNameControl(FILE *fd,int amount,char* rootDir);**
- **void rootDirControl(FILE *fd,int amount,char* rootDir);**
- **char* readFile(FILE* fp,int length);**
- **void updateDataInFSM(FILE *fd,int fsmStart);**
- **void updateNodeInFSM(FILE* fd,int fsmStart);**
- **void listRootDir(FILE*fd);**

MY CONTOLS ABOUT INVALID OPERATION ATTEMPTS:

- For invalid paths,all operations prints error.
- If mkdir operation is making with same name directory in parent directory,prints error.
- If write operation is making with same name file in parent directory,prints error.
- Missing and more commanline argments,prints error.

NOTE:According to my file system,length of file and directory names can be at most 30. Otherwise error message is printing.

Nevra Gürses
161044071

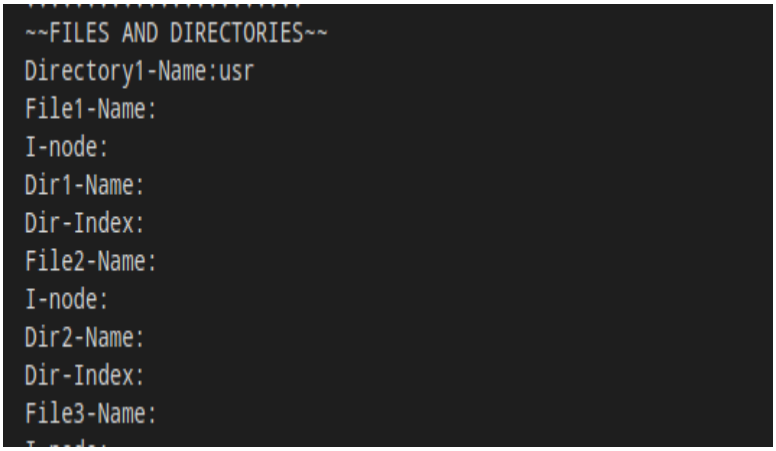
RUNNING RESULTS:

MKDIR OPERATION:

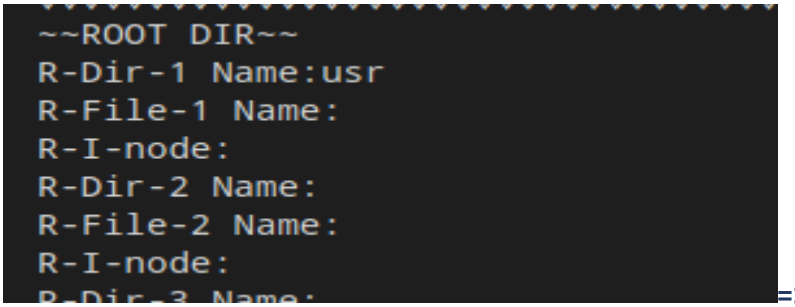
Input:

```
cse312@ubuntu:~/Desktop$ gcc fileSystemOper.c -o fileSystemOper -Wall -Wextra
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data mkdir "/usr"
```

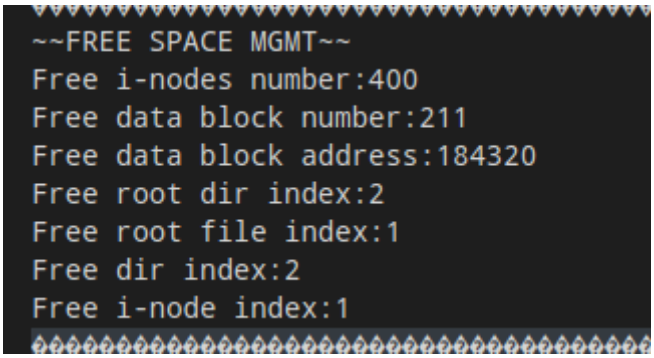
Outputs:



=>Files and directories part.



=>Root dir part.



(Free space management part is updated.)

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data mkdir "/usr/nevra"
```

Outputs:

```
~~FILES AND DIRECTORIES~~
Directory1-Name:usr
File1-Name:
I-node:
Dir1-Name:nevra
Dir-Index:1
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
Dir3-Name:
Dir-Index:
File4-Name:
I-node:
Directory2-Name:nevra
File1-Name:
I-node:
Dir1-Name:
Dir-Index:
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
```

(Inner directory named nevra is created in directory usr.And new directory place for nevra directory allocated in files and directories section.)

```
~~FREE SPACE MGMT~~
Free i-nodes number:400
Free data block number:211
Free data block address:184320
Free root dir index:2
Free root file index:1
Free dir index:3
Free i-node index:1
```

(Free space management part is updated)

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data mkdir "/bin/nevra"
Error!No such directory in rootDir->bin
```

(Because of no directory named bin , error is printing.)

Nevra Gürses
161044071

WRITE OPERATION:

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data write "/usr/nevra/file1" input.txt
```

Outputs:

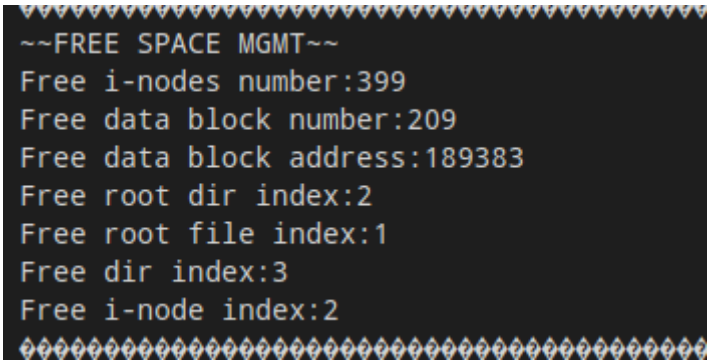
```
~~I-NODES~~
I-node:1
Name of file:file1
Last modification date and time:Sun May 31 09:48:51 2020
Size of file:5047
Disk-0 Address:184336
Disk-1 Address:188432
Disk-2 Address:
Disk-3 Address:
Disk-4 Address:
Disk-5 Address:
Single Indirect Block:
Double Indirect Block:
Triple Indirect Block:
```

(File is pointed in first free i-node.Size of input file is bigger than 1 block size,so 2 disk address allocated fort his file.)

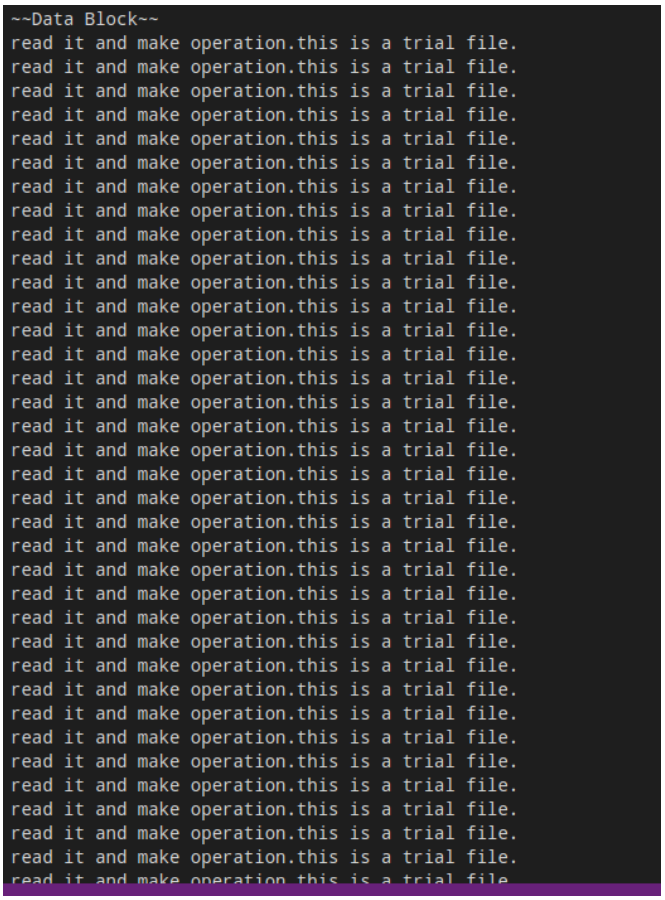
```
~~FILES AND DIRECTORIES~~
Directory1-Name:usr
File1-Name:
I-node:
Dir1-Name:nevra
Dir-Index:1
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
Dir3-Name:
Dir-Index:
File4-Name:
I-node:
Directory2-Name:nevra
File1-Name:file1
I-node:1
Dir1-Name:
Dir-Index:
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
```

(File is created in inner directory named nevra)

Nevra Gürses
161044071



(Free space management updated.)



(File is copied in data block section.)

Nevra Gürses
161044071

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data write "/usr/file2" input.txt
```

Outputs:

```
I-node:2
Name of file:file2
Last modification date and time:Sun May 31 09:54:26 2020
Size of file:5047
Disk-0 Address:189399
Disk-1 Address:193495
Disk-2 Address:
Disk-3 Address:
Disk-4 Address:
Disk-5 Address:
Single Indirect Block:
Double Indirect Block:
Triple Indirect Block:
```

(File is pointed second i-node.Size of input file is bigger than 1 block size,so 2 disk address allocated for this file.)

```
~~FILES AND DIRECTORIES~~
Directory1-Name:usr
File1-Name:file2
I-node:2
Dir1-Name:nevra
Dir-Index:1
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
Dir3-Name:
Dir-Index:
File4-Name:
I-node:
Directory2-Name:nevra
File1-Name:file1
I-node:1
Dir1-Name:
Dir-Index:
File2-Name:
I-node:
Dir2-Name:
Dir-Index:
File3-Name:
I-node:
```

(File is created in directory named usr.)

```
~~FREE SPACE MGMT~~
Free i-nodes number:398
Free data block number:207
Free data block address:194446
Free root dir index:2
Free root file index:1
Free dir index:3
Free i-node index:3
```

=>FSM is updated.

Nevra Gürses
161044071

```
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
```

(File is copied in data block section.)

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data write "/file3" input.txt
cse312@ubuntu:~/Desktop$
```

Ouputs:

```
Name of file:file3
Last modification date and time:Sun May 31 09:59:15 2020
Size of file:5047
Disk-0 Address:194462
Disk-1 Address:198558
Disk-2 Address:
Disk-3 Address:
Disk-4 Address:
Disk-5 Address:
Single Indirect Block:
Double Indirect Block:
Triple Indirect Block:
```

(File is pointed by third i-node.Size of input file is bigger than 1 block size,so 2 disk address allocated for this file.)

```
~~ROOT DIR~~
R-Dir-1 Name:usr
R-File-1 Name:file3
R-I-node:3
R-Dir-2 Name:
R-File-2 Name:
R-I-node:
R-Dir-3 Name:
```

(File is writed in root directory.)

```
~~FREE SPACE MGMT~~
Free i-nodes number:397
Free data block number:205
Free data block address:199509
Free root dir index:2
Free root file index:2
Free dir index:3
Free i-node index:4
```

(Free space management part is updated.)

```
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
```

(File is copied in data block section.)

LIST OPERATION:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data list "/"
usr

file3
```

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data list "/usr"
file2

nevra
```

Nevra Gürses
161044071

READ OPERATION:

Input:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data read "/usr/file2" out.txt
```

Ouput:

```
home ▸ cse312 ▸ Desktop ▸ ≡ out.txt

read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
read it and make operation.this is a trial file.
```

(Out.txt file)

DUMPE2FS OPERATION:

```
cse312@ubuntu:~/Desktop$ ./fileSystemOper fileSystem.data dumpe2fs
Block Size:4096
Block Count:256
I-node Block Count:38
I-node Count:400
Number of Directories:36
Number of Files:400
~~~Occupied I-Node Blocks and File Names In That Blocks~~~
I-node:1
file1
I-node:2
file2
I-node:3
file3
cse312@ubuntu:~/Desktop$
```