FILE SIMULATION
OPERATING SYSTEM PRINCIPLES
COSC (519)

SUBMITED TO
Professor.RAMESH K.KARNE.

1.FILE SIMULATION

INTRODUCTION

- File system is the process through which we can save and arrange the different files and their data in the computer system. It also helps the user to access, search and retrieve the data from the file.
- We can also perform the same functions on the files also. The project which we have chosen can be implemented in the real world.
- The main aim of this project is that we can build such a simulator which can be helpful in treating the virtual file system as commands in the operating system.
- With the help of this project we can make many changes in the data which is being stored in the files.
- For example we can create, alter, add and delete files and directories as well .
- It also provides some extra features like access control, file permissions and attribute enhancement to a particular file of a system.

Approach:

We will use C++ language to simulate the file management system. Our file system will have following properties.

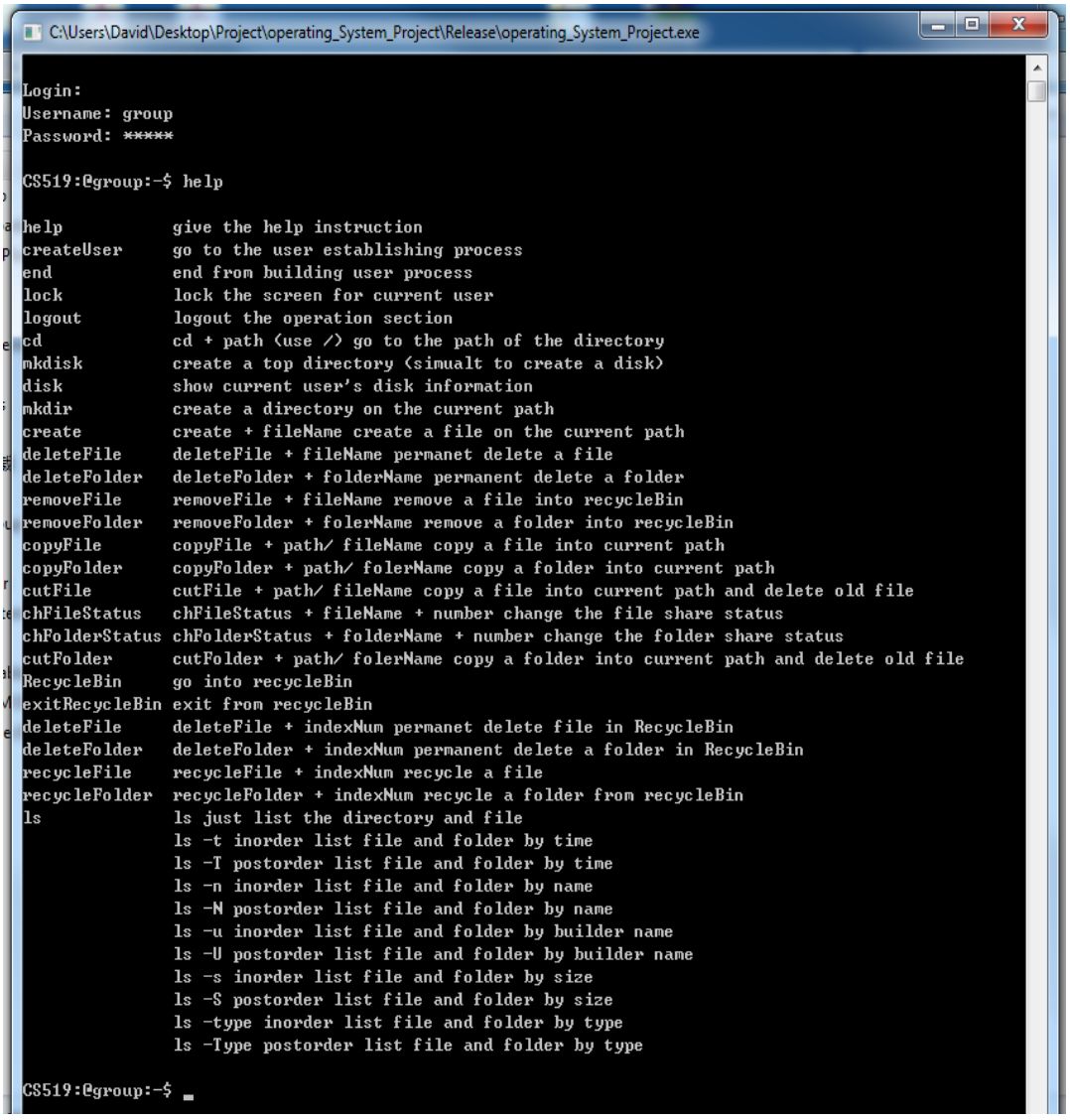
1. Multi-user file system
2. Multi-level directory structures
3. Locking mechanism technique
4. Security for multiple users

- First of all, our file management system is designed for multiple users.
- Secondly, we will create multi-level directory and files to invent our own file system.
- We choose sequence directory, because it is more efficient and easier to reach the target file by running the search function.
- We can also try to create and delete files in a hierarchical data organization.
- Thirdly, we will use locking mechanism to prevent the situation in which more than one user access into the same file.
- Finally, we will specify a file security descriptor for a file or directory when we are running create file or directory functions.

2.OPERATIONS INCLUDED IN FILE SYSTEM

In order to define a file properly, the operations that can be performed should be considered. The six basic file operations for a file, which are supported by the operating system by providing system calls, are given below:

- Execute the help command then it will retrieve the commands in the File System .
  COMMAND- cs519:@group:-$ help

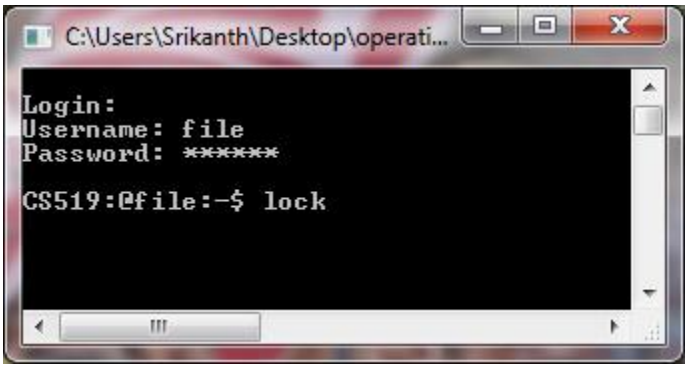


- To create the disk use the command **mkdisk**
- When a file is closed, its entry is removed from open file table

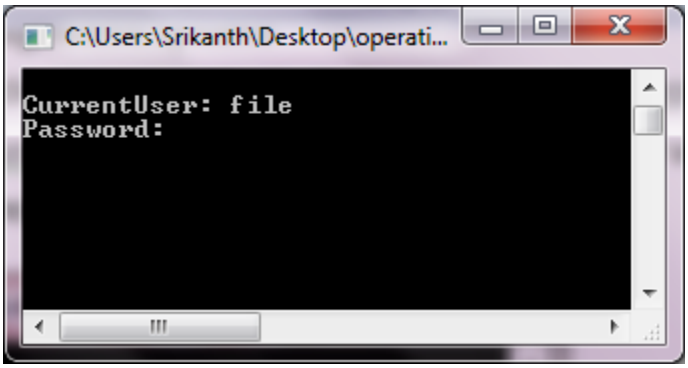
3.SECURITY ISSUE IN THE FILE SYSTEM(ACCESS & PERMISSION)

- Here In this File management system the user can not perform the operations directly .
- In order to perform the operations the user needs to create the login and password.
- Once when the user enter the ID and password user need to type the **end Command .**
- Once if he created the ID and password now he is allowed to access the file system
- The user can use the **lock** commend to protect his data on the file management system
- In order to open the lock the user need to type the id and password again.
- The user can logout once if he complete the work.

  Figure to explain the above data



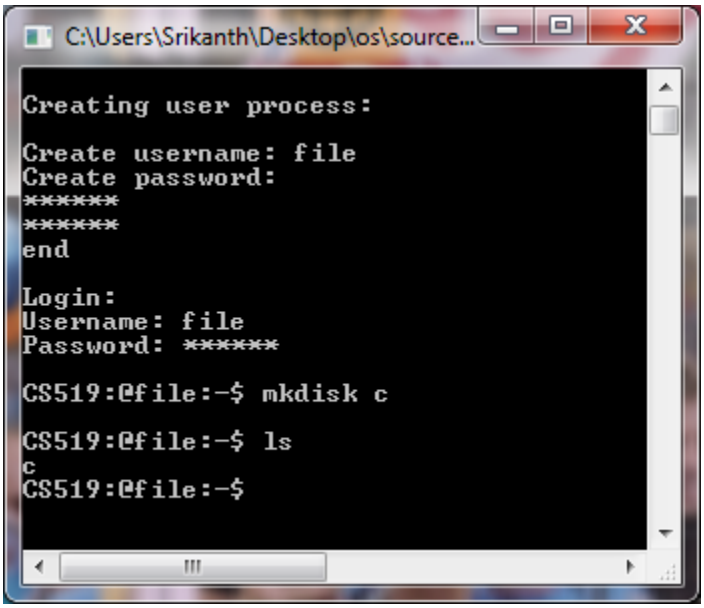
- After the lock key is used then the function is going to be shown in the below Diagram



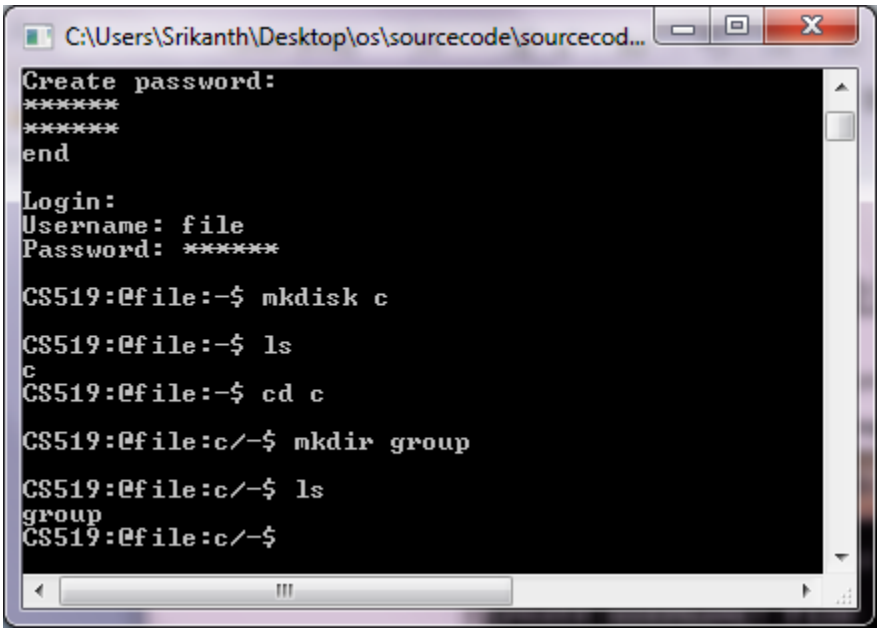
4.DIRECTORY AND FILE STRUCTURE

FUNCTIONALITY'S USED IN THE FILE SYSTEM

- **CREATE A DISK -mkdisk** is a command used to create a disk.
  The commend is **mkdisk disk name**

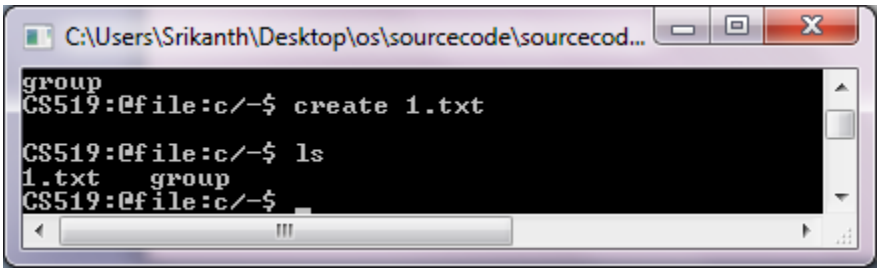


- **CREATE A DIRECTORY – mkdir** is the command used to create the directory.
  The command used is **mkdir dir name**

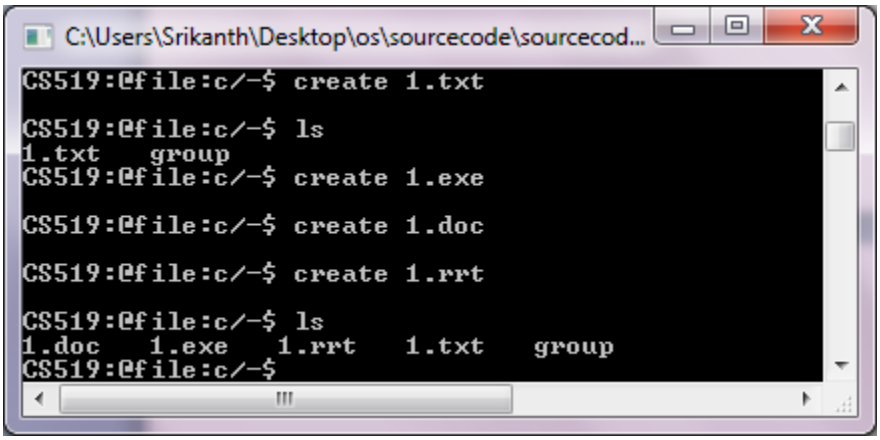


  The user can create number of directory's like this.

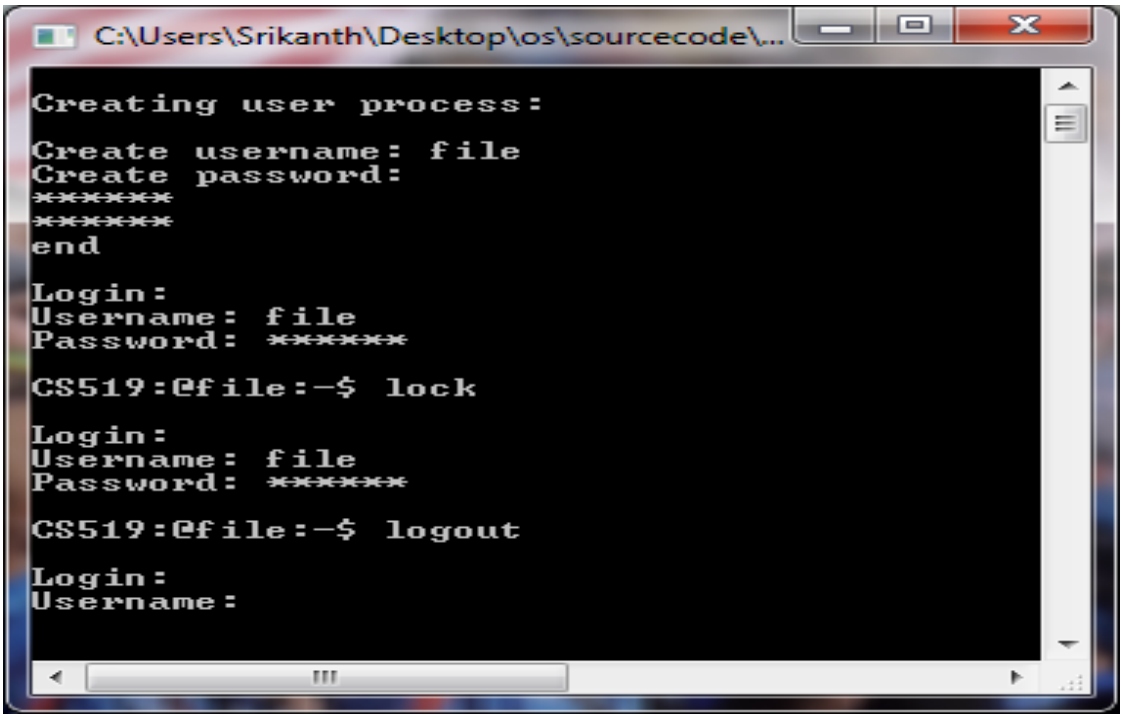
- **CREATE FILE –create** is the command to create the file in side the directory.
  The command used for the create the file is **create filename**



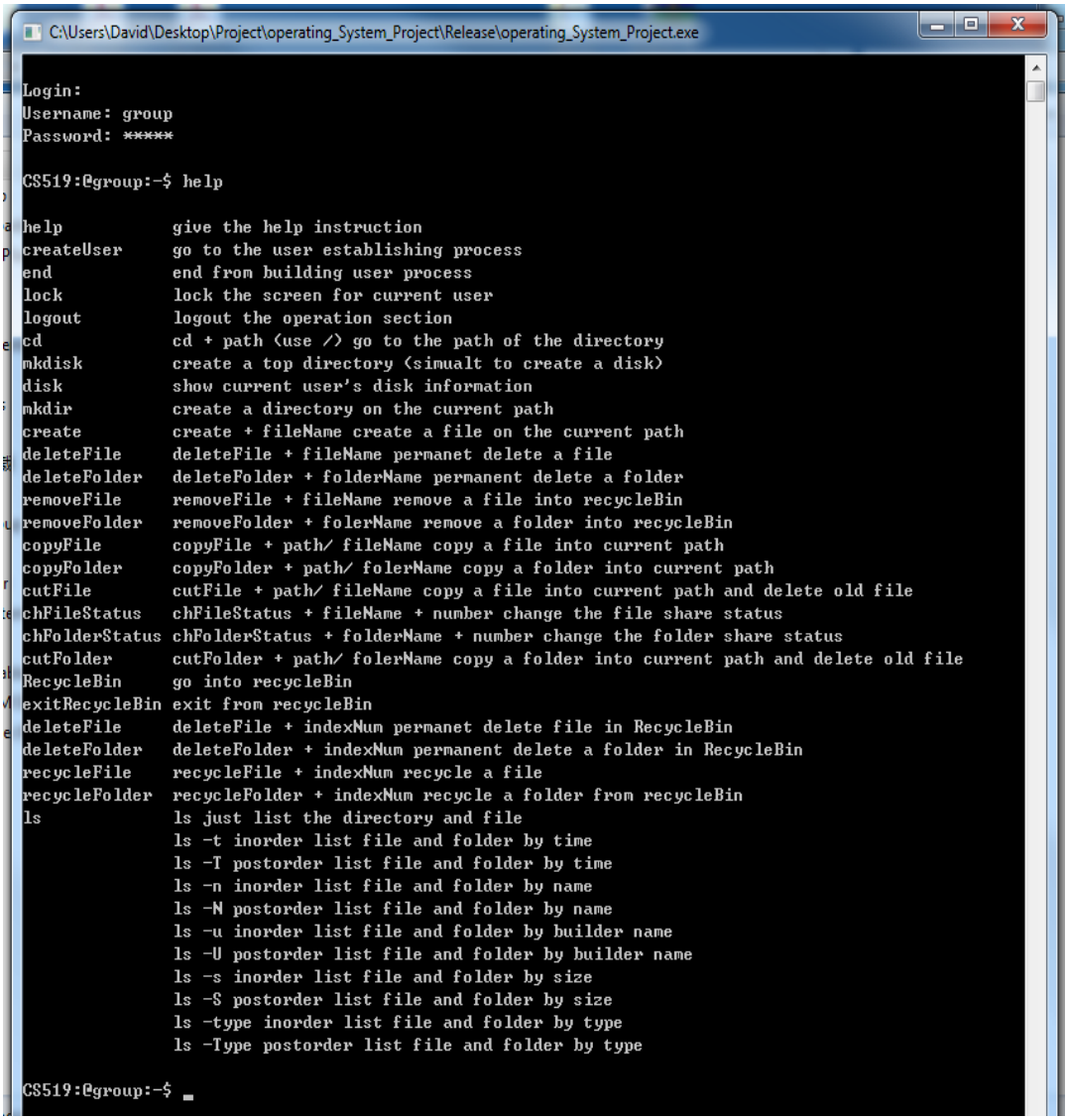
Here in the above diagram we can give the file type like txt,exe,doc,rrt etc



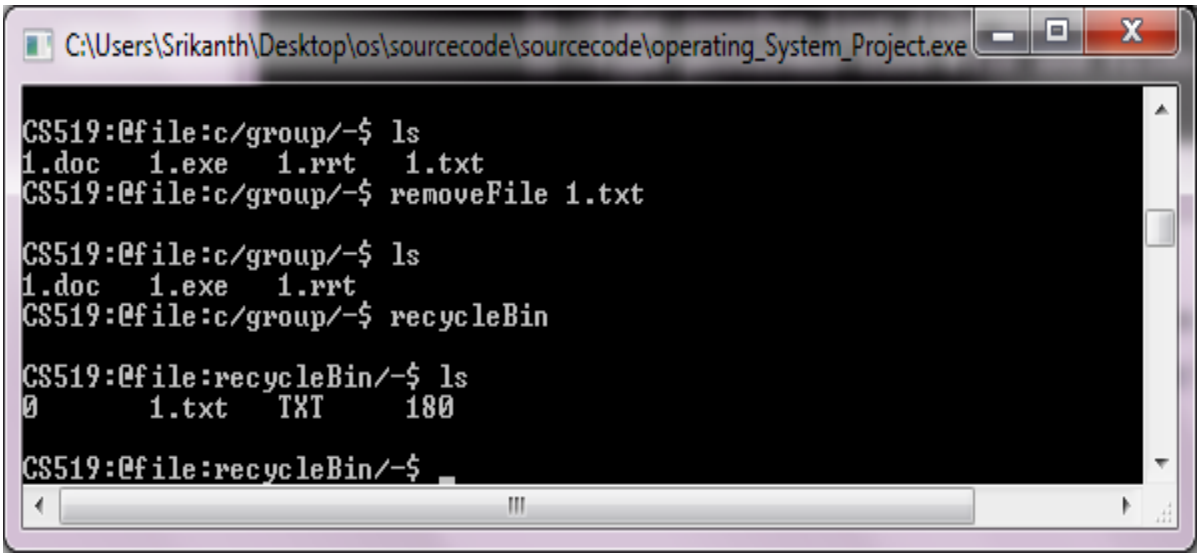
- **LOGIN IN TO THE FILE SYSTEM**
- **To LOCK THE FILESYSTEM**
- **TO LOGOUT OF THE FILE SYSTEM**



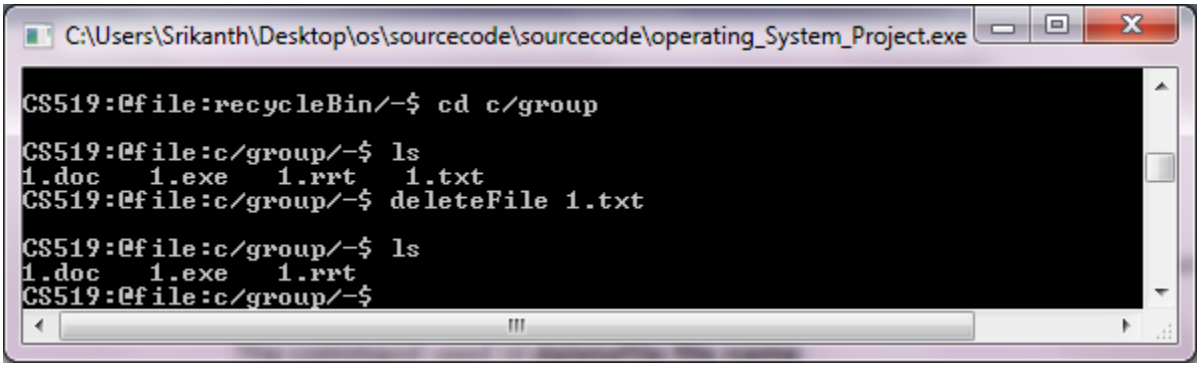
- **HELP-help** is the command used to display the commands in the file system



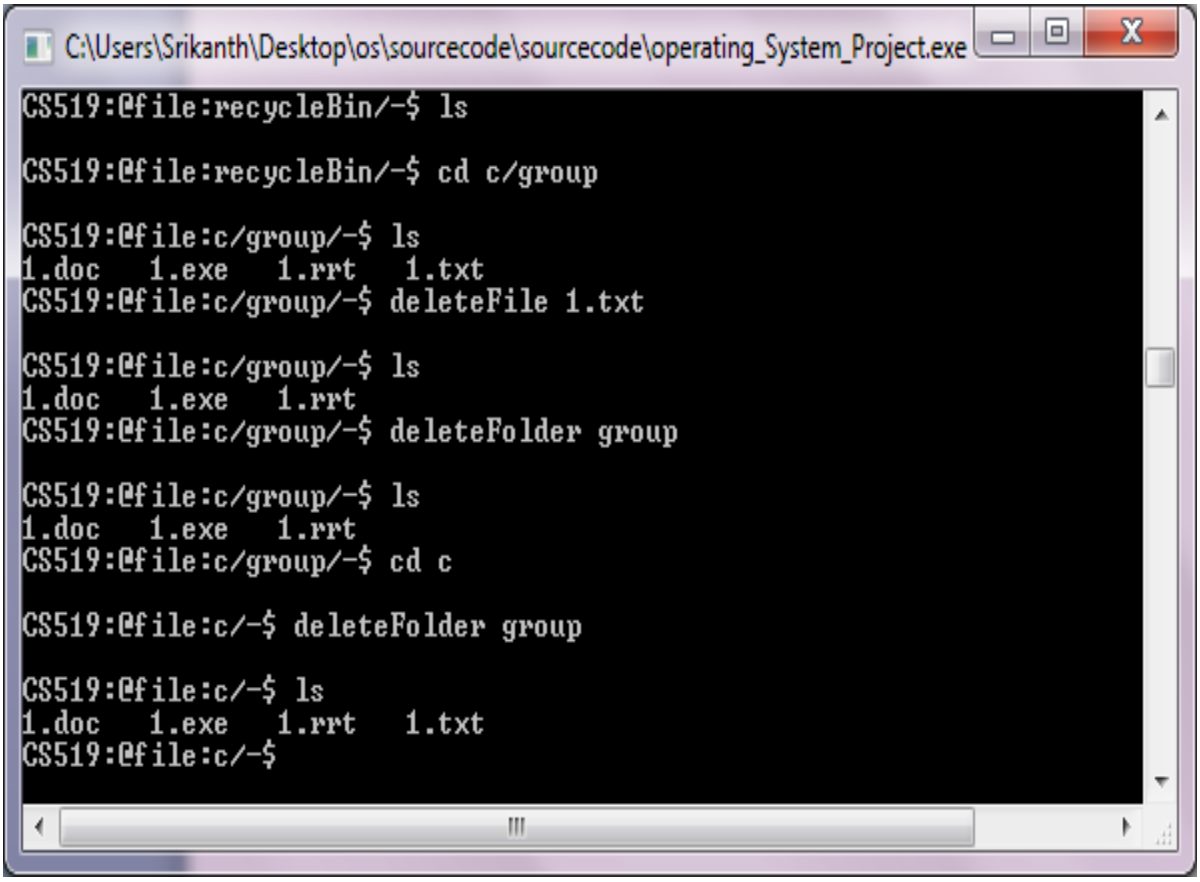
- **TO REMOVE FILE-removeFile** is a command used to remove the file from the directory and stores in the recycleBin .If we want the removed file we can restore that file from the recycleBin



- **TO DELETE FILE –deleteFile** is used to delete the file from the directory and we can not recycle the that file again
  The command used id **deleteFile file name**

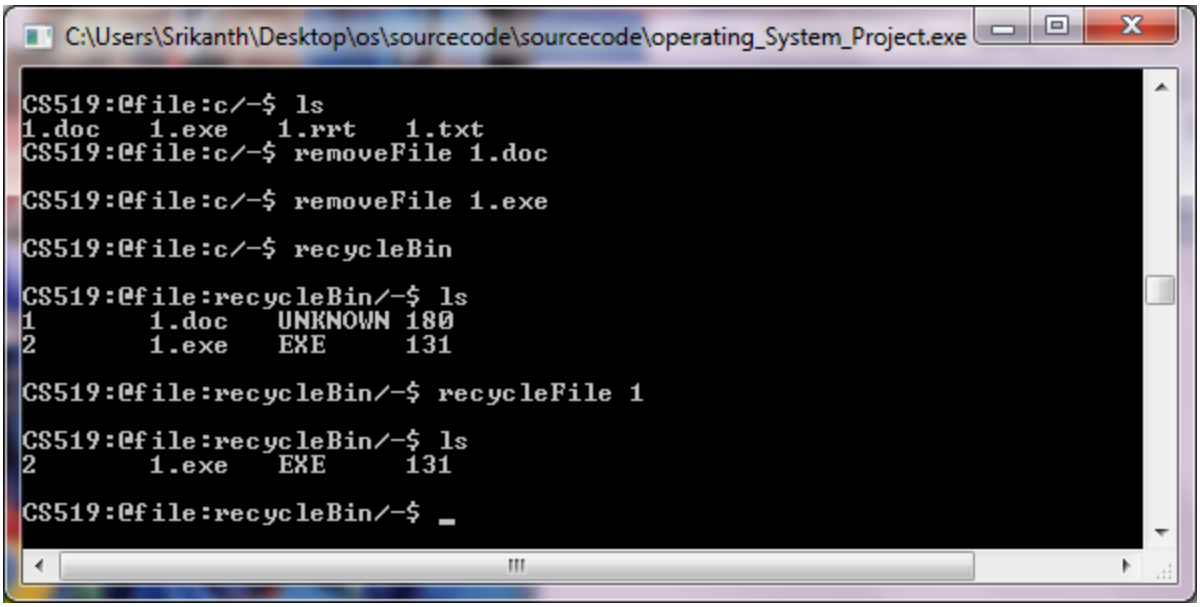


- **REMOVE DIRECTORY- removeDirectory is used to remove the directory**
- **Delete DIRECTORY- deleteDirectory is used to delete the Directory**

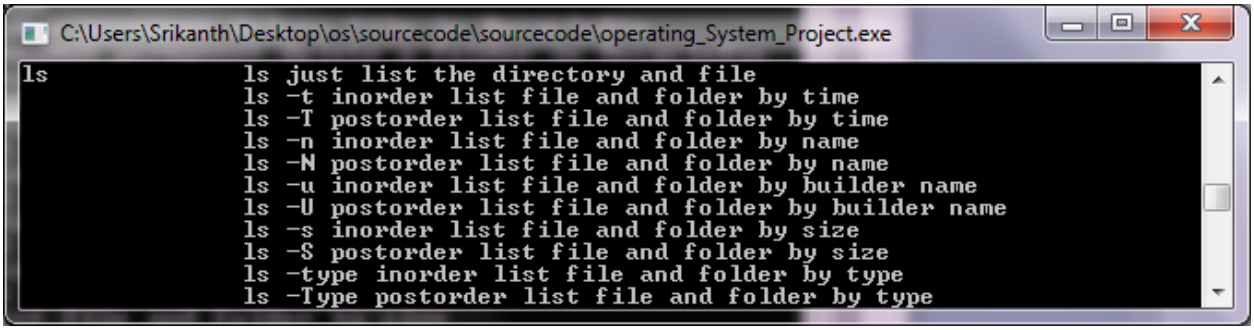


- **RecycleBin-recyclebin is use to store the removed files**
- **RecyclrFile –is used to restore the file back.**
- **RecycleFolder- is used to restore the folder.**

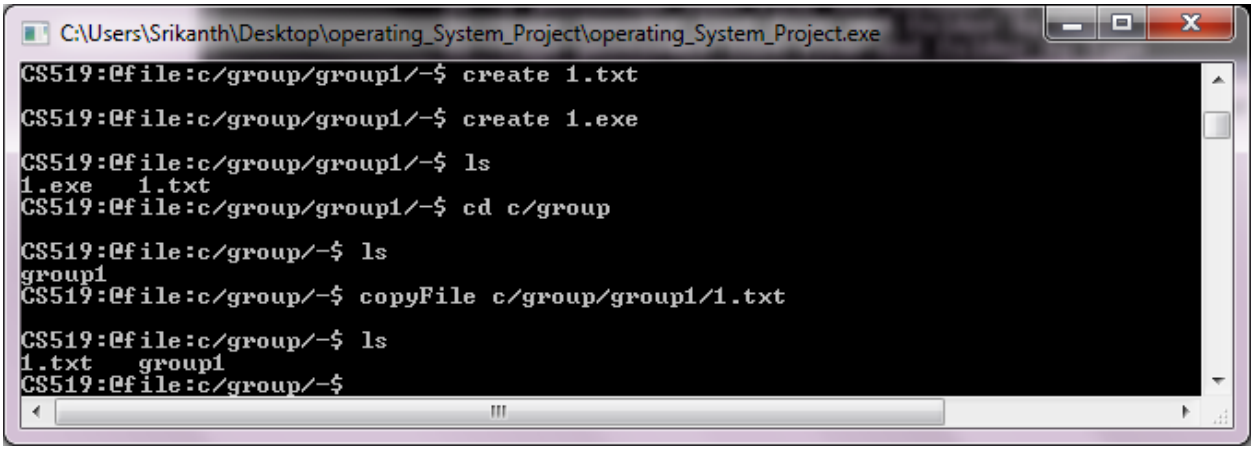
  Note- when you want to recycle something we need to give memory address number which is in the recycleBin.



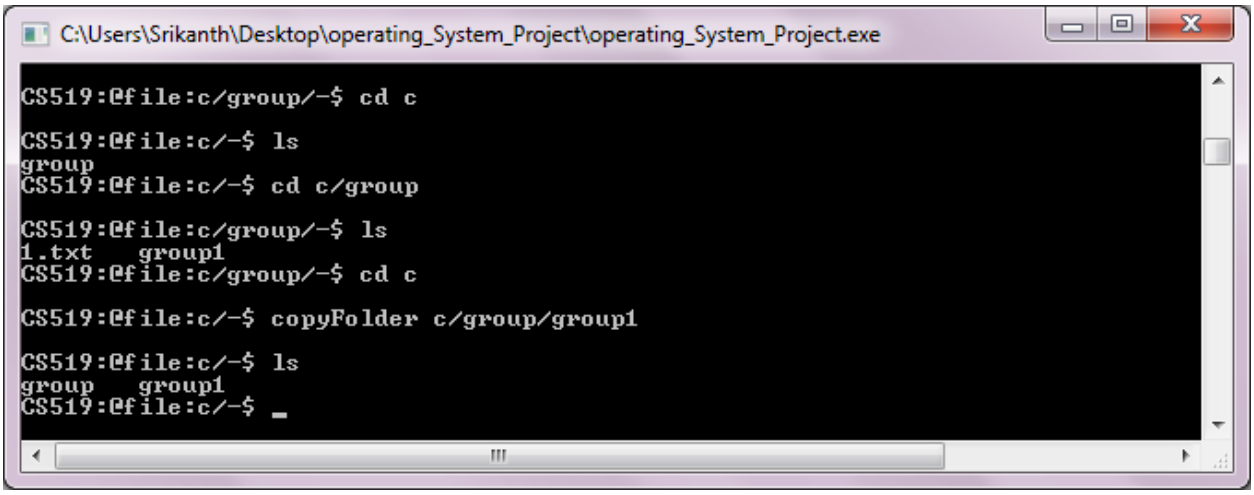
- TO DISPLAY –ls is the command used to display
- The Different type of **LS functions**



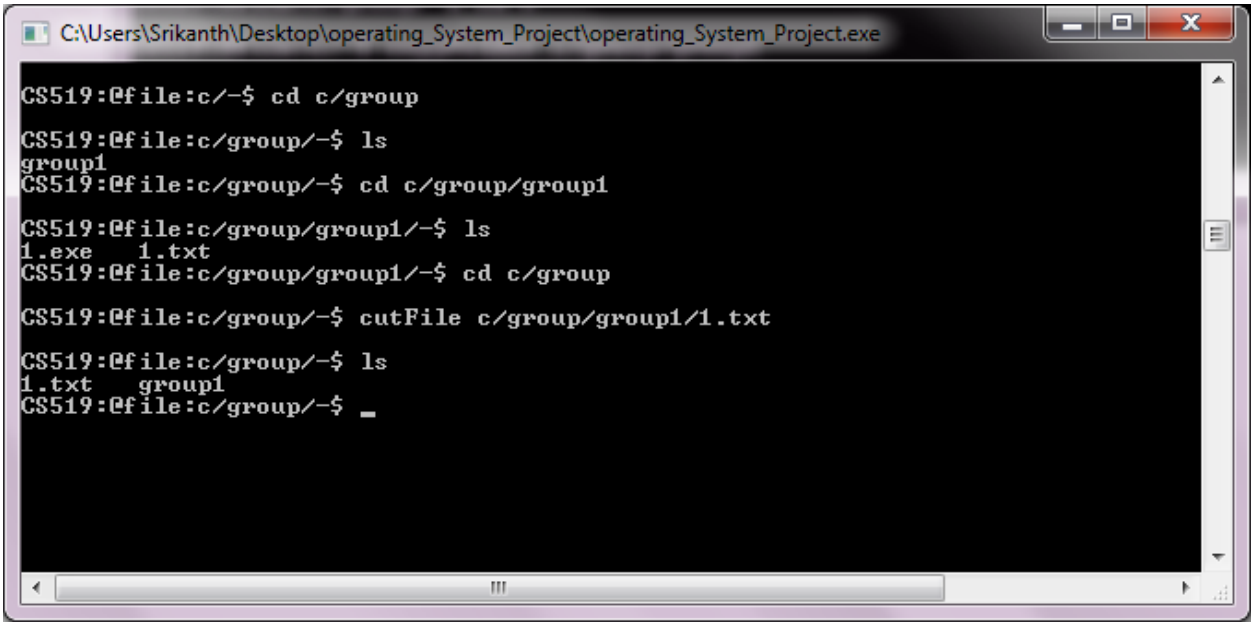
- **TO COPY FILE –copyFile** is the command for to copy the file from one place to another place.
  The command for to copy the file is **copyFile path+file name to be copied**



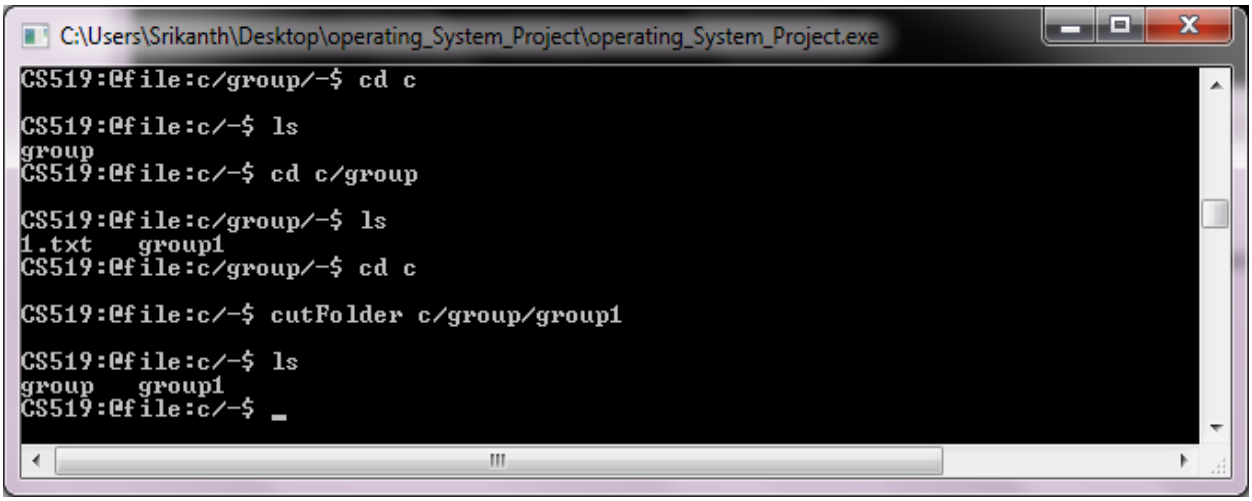
- **TO COPY FOLDER –copyFolder** is the command for to copy the folder from one place to another place.
  The command for to copy the file is **copyFolder path+file name to be copied**



- **TO CUT FILE –cutFile** is the command for to cut the file from one place to another place.
  The command for to cut the file is **cutFile path+file name to be cut**



- **TO CUT FOLDER –cutFolder** is the command for to cut the file from one place to another place.
  The command for to cut the file is **cutFolder path+file name to be cut**



5.DESCRIPTION of THE PROGRAMING

CLASS FOR FUNCTION DECLERATON

public fileTree()- Here in the project this function will gives the information about the structure used for the file implementation

- The variables used in this file tree class is

```
tree_node* left;
tree_node* right;
file data;
tree_node* root;
tree_node* recycle_root;
```

class file()- this is the class to give the description about the file function implementation

- The variables used in this file class is

Here in this class we created one file constructer and we will add the functions to that constructer EX- `file();`

```
file(const string& na);
```

class fDataTable- Here in this fDataTable we will define the data table structure

- The variables used in this class is

```
int dataBegin;
int dataEnd;
int size;
```

class fDataTree-Here in this function we are defining the file data storage structure.

- The variables used in this class is

```
struct tree_node
	{
		tree_node* left;
		tree_node* right;
		const file* data;
	};
```

class fData

```
{
private:
	vector<fDataTable> data;
	int dataSize;
public:
	fData();
	fData(const vector<fDataTable>& t);
	~fData();

	int getDataSize() ;
```

class folder:public file

- This is the function class for the directory where the file should be in said the folder.
- Here in this class we will import all the function s of the file and we will implement the file here as the child of the folder.

```
	#ifndef _folder_
	#define _folder_

	#include "file.h"
	#include "fileTree.h"
	#include "fTypeTree.h"
	#include "fTimeTree.h"
	#include "fDataTree.h"
	#include "fShareTree.h"


	class folder:public file
	{
	protected:
		fileTree indexFile;
		fTypeTree indexfType;
		fTimeTree indexfTime;
		fDataTree indexfData;
		fShareTree indexfShare;

	protected:
		void calDataSize();
	public:
		folder();
```

class folderTree

- Here we implemented Binary Tree structure for folder tree .
- A Binary tree is the one where the file with small size is added to the left side of the tree
- If the size of the other tree as more size then it will added to the right side of the tree
- The variables used in this class is

```
struct tree_node
	{
		tree_node* left;
		tree_node* right;
		folder data;
		tree_node* FirstChildren;
		tree_node* rootParent;
	};
```

class fType

- This is the class where its going to be used for different file types
- Like file.txt,file.doc,file.exe etc
- The variables used in this class is

```
#define UNKNOWN 0;
#define EXE 1;
#define DAT 2;
#define TXT 3;
#define DOCX 4;
#define PPTX 5;
#define XLSX 6;
#define FOLD 7;
#define DISK 8;

class fType
{
private:
	fileType type;
	int linkExe;
```

class fTypeTree

- This is the class which give the structure for file type storage .
- The main Structure used here is

```
#include "file.h"
class fTypeTree
{
private:
	struct tree_node
	{
		tree_node* left;
		tree_node* right;
		const file* data;
	};
```

class recycle

- this is the class for the RecycleBin
- when the remove command id used one a file or folder then those ones are going to store in the RecycleBin

class recycleTree

- This is the class which gives the storage structure for the RecycleBin
- Function types declaration

```
#include "recycle.h"

class recycleTree
{
private:
	struct tree_node
	{
		tree_node* left;
		tree_node* right;
		recycle data;
	};
	tree_node* root;
```

class fTime

- This is the function for the time displaying function
- By using this function we can access the time for when the things are created

class fTimeTree

- This is the class which gives the storage structure for the time

Function types declaration

class fUser-

- This is the function used for the implementation of the user login module.
- This module is the one which allows the user to use file management system.

class userTree

- This function will gives the structure for the user login module.

class block:public folder

- This is the which will gives the tree structure for the project
- This function is always connected to the folder
- The tree structure we implemented here is binary tree

Class blockTree-this class maintains the file structure for the block class.

extern_Variable.h-This will give the size of the file and folder.

class fShare and class fShareTree – This two classes are used for the implementation of the read and write permissions .Simply this class are used for the maintain the authentication of the file management.

class frame

- This is the main class which will implement all the functions in this class like it will gives the connection for all functions.

Function.h

- This is the heeder file used for the to accept the string when the user creating the account in the file system
- This heeder file give all the details in help function.
- By using this help file every one can understand the file operation available to the user.