FAT SYSTEM

Version 1.0 approved

Prepared by

1. Mahmoud Khairy
2. Mohamed Sayed
3. Mahmoud Ahmed

DATA: <6 May. 22>

The Project have 4 Classes (Core)

First Class: `VirtualDisk`

Function 1	initialize
Input	string path
Output	
Processing	Check if file of fat exist or not If not found Create file and write file cluster (1024) bits And prepare Fat (array of intger) size [1024] Make ROOT directory and make this the current program Then write in Fat

Source Code	`public static void initialize(string path)`
	`{`
	`if (!File.Exists(path))`
	`{`
	`Disk = new FileStream(path, FileMode.OpenOrCreate, FileAccess.ReadWrite);`
	`byte[] b = new byte[1024];`
	`for (int i = 0; i < b.Length; i++)`
	`b[i] = 0;`
	`writeCluster(0,b);`
	`FAT.prepareFat();`
	`Directory root = new Directory("M:", 0x10, 5, null);`
	`root.WriteDirectory();`
	`Program.current = root;`
	`FAT.writeFat();`
	`}`
	`else`
	`{`
	`Disk = new FileStream(path, FileMode.OpenOrCreate, FileAccess.ReadWrite);`
	`FAT.readFat();`
	`Directory root = new Directory("M:", 0x10, 5, null);`
	`root.ReadDirectory();`
	`Program.current = root;`
	`}`
	`}`

Function 2	`readCluster`
Input	`int clusterIndex`
Output	`byte[] b`
Processing	Put seek begin on clusterIndex * 1024 And read on array of byte from Disk (File Created)

Source Code	`public static byte[] readCluster(int clusterIndex)`
	`{`
	`Disk.Seek(clusterIndex * 1024,`

	SeekOrigin.Begin);
	byte[] b = new byte[1024];
	Disk.Read(b, 0,1024);
	return b;
	}

Function 3	writeCluster
Input	int clusterIndex, byte[] bytes
Output	Void
Processing	Put seek begin on clusterIndex * 1024 And write an of byte from 0 to length of array

Source Code	public static void writeCluster(int clusterIndex, byte[] bytes)
	{
	Disk.Seek(clusterIndex * 1024, SeekOrigin.Begin);
	Disk.Write(bytes, 0, bytes.Length);
	Disk.Flush();
	}

Second Class: FAT

Function 1	prepareFat
Input	none
Output	void
Processing	Loop for length of array and put: index 0 and 4 value 0 index between 0 and 4 value index + 1 Else put value of them 0
Source Code	public static void prepareFat()

	{
	
	for (int i = 0; i < fat.Length; i++)
	{
	if (i == 0 || i == 4)
	{
	fat[i] = -1;
	}
	else if (i > 0 && i <= 3)
	{
	fat[i] = i + 1;
	}
	else
	{
	fat[i] = 0;
	}
	}
	}//prepareFat

Function 2	writeFat
Input	none
Output	void
Processing	Make array of byte length equal fat.length * 4 And convert array of intger to array of byte then split array of byte clusters 1024 (array) and put all of them in list of array Then loop on count of this list and write cluster all of them in Disk File

Source Code	public static void writeFat()
	{
	byte[] bytes = new byte[fat.Length * sizeof(int)];
	
	Buffer.BlockCopy(fat, 0, bytes, 0, bytes.Length);
	
	List<byte[]> ls = VirtualDisk.splitBytes(bytes);
	
	for (int i = 0; i < ls.Count; i++)
	{
	VirtualDisk.writeCluster(i + 1, ls[i]);
	}
	
	}//writeFat

Function 3	readFat
Input	none
Output	void
Processing	Make buffer (array of byte) size (4096) first 4 cluster And loop of them cluster and put of them in list of array of byte Make array of byte called bytes and covert list of array of byte and put on “bytes” and covert array of bytes to array of intger by method Buffer.BlockCopy

Source Code	`public static void readFat()`
	`{`
	`byte[] buffer = new byte[4096];`
	`List<byte> ls = new List<byte>();`
	
	`for (int i = 0; i <4 ; i++)`
	`{`
	`ls.AddRange(VirtualDisk.readCluster(i + 1));`
	`}`
	
	`byte[] bytes = ls.ToArray();`
	`int[] fatAsIntegers = new int[bytes.Length / sizeof(int)];`
	
	`Buffer.BlockCopy(bytes, 0, fatAsIntegers, 0, bytes.Length);`
	
	`fat = fatAsIntegers;`
	
	`}//readFat`

Function 4	GetEmptyCulster
Input	none
Output	Index of empty cluster
Processing	Loop for fat length And check if index of them value equal 0 Return the index Else return -1

Source Code	`public static int GetEmptyCulster()`
	`{`
	`for(int i = 0; i < fat.Length; i++)`
	`{`
	`if(fat[i] == 0)`
	`{`
	`return i;`
	`}`
	`}`
	`return -1;`
	`}`

Function 4	`getAvailableClusters`
Input	none
Output	Count of Availabele Clusters
Processing	Loop for fat length And check if index of them valued equal 0 Increase count++ After loop Return Count

Source Code	`public static int getAvailableClusters()`
	`{`
	`int count = 0;`
	`for (int i = 0; i < fat.Length; i++)`
	`{`
	`if (fat[i] == 0)`
	`count++;`

	} return count; }

Third Class: Directory: DirectoryEntry (inheritance)

Has
-public Directory parent -> refer to the parent of directory
-public list DirectoryEntry(entries) -> refer to all files/Direcotry in this Directory

Function 1	searchDirectory
Input	string name
Output	Index of Directory in entries
Processing	First ReadDirecotry Then substring 11 char of string name Then for loop in all entries Get name of all entries and equal with string name If found return index of this entry Else return -1

Source Code	public int searchDirectory(string name)
	{
	ReadDirectory();
	if (name.Length < 11)
	{
	name += "\0";
	for (int i = name.Length + 1; i < 12; i++)
	name += " ";
	}
	else
	{
	name = name.Substring(0, 11);
	}
	for (int i = 0; i < entries.Count; i++)
	{
	string Dname = new string(entries[i].dir_name);
	if (Dname.Equals(name))
	return i;
	}
	return -1;
	}

Function 2	addEntry
Input	DirectoryEntry d (object of directory)
Output	void
Processing	Add this object in entries And write directory

Source Code	public void addEntry(DirectoryEntry d)
	{
	entries.Add(d);
	WriteDirectory();
	}

Function 3	ReadDirectory
Input	none
Output	void
Processing	Check first_cluster not equal zero Make list of entries This add range in list of array of byte read cluster until next equal -1 then out of this loop After that Make loop of ls count And nake array of byte size 32 byte And assign from ls to small array Then add this in entries after makeDirectory

Source Code	`public void ReadDirectory()`
	`{`
	`if (this.firs_cluster != 0)`
	`{`
	`entries = new List<DirectoryEntry>();`
	`int cluster = this.firs_cluster;`
	`int next = FAT.getClusterNext(cluster);`
	`List<byte> ls = new List<byte>();`
	`do`
	`{`
	`ls.AddRange(VirtualDisk.readCluster(cluster));`
	`cluster = next;`
	`if (cluster != -1)`
	`next = FAT.getClusterNext(cluster);`
	`}`
	`while (next != -1);`
	`for (int i = 0; i < ls.Count; i++)`
	`{`
	`byte[] b = new byte[32];`
	`for (int k = i * 32, m = 0; m < b.Length && k < ls.Count; m++, k++)`
	`{`
	`b[m] = ls[k];`
	`}`
	`if (b[0] == 0)`
	`break;`
	`entries.Add(makeDirectory(b));`
	`}`
	`}`
	`}`

Function 4	WriteDirectory
Input	none
Output	void
Processing	Loop on entries count And take index of them and convert in byte (32 byte) And assign them take in account offest in big array of byte called DirorFilesinBytes After that split big array in list of array byte 1024 called bytesls Check first_cluster not equal zero

	Put clusterindex the value of fisrt_cluster
	If not put clusterindex value FAT.GetEmptyCulster();
	Loop in bytesls
	Check fully disk or not
	Then write cluster of all index of list
	Put clusternext equal -1 in this index
	Then check if (lastCluster != -1)
	FAT.setClusterNext(lastCluster,clusterFATIndex);
	First lastCluster equal -1 by defalut then not access this if
	
	lastCluster = clusterFATIndex;
	Then put lastCluster value of index of cluster
	clusterFATIndex = FAT.GetEmptyCulster();
	And new of cluster index get empty cluster in Fat
	
	If entries count eqaul zero then no data found
	To write then put first cluster eqaul 0
	
	If this directory has parent (not NULL)
	Update content of parent
	Then call fun writedirecotry again
	And finally writeFat

Source Code	`public void WriteDirectory()`
	`{`
	`byte[] dirsorfilesBYTES = new byte[entries.Count * 32];`
	`for (int i = 0; i < entries.Count; i++)`
	`{`
	`byte[] b = DirToByte(this.entries[i]);`
	`for (int j = i * 32, k = 0; k < b.Length; k++, j++)`
	`dirsorfilesBYTES[j] = b[k];`
	`}`
	`List<byte[]> bytesls = VirtualDisk.splitBytes(dirsorfilesBYTES);`
	`int clusterFATIndex;`
	`if (this.firs_cluster != 0)`
	`{`
	`clusterFATIndex = this.firs_cluster;`
	`}`
	`else`
	`{`
	`clusterFATIndex = FAT.GetEmptyCulster();`
	`this.firs_cluster = clusterFATIndex;`
	`}`
	`int lastCluster = -1;`
	`for (int i = 0; i < bytesls.Count; i++)`
	`{`
	`if (clusterFATIndex != -1)`
	`{`
	`VirtualDisk.writeCluster(clusterFATIndex,bytesls[i]);`
	`FAT.setClusterNext(clusterFATIndex, -1);`
	`if (lastCluster != -1)`
	`FAT.setClusterNext(lastCluster, clusterFATIndex);`
	`lastCluster = clusterFATIndex;`
	`clusterFATIndex =`

	FAT.GetEmptyCulster();
	}
	}
	if (entries.Count == 0)
	{
	if (firs_cluster != 0)
	FAT.setClusterNext(firs_cluster, 0);
	firs_cluster = 0;
	}
	if (this.parent != null)
	{
	
	this.parent.updateContent(this.getDirectoryEntry());
	this.parent.WriteDirectory();
	}
	FAT.writeFat();
	}

Function 5	deleteDirectory
Input	none
Output	void
Processing	ClearDirSize; Check if the parent of this direcotry not null Search on dir_name of program current If found Parent ReadDirectory Remove from parent entries this index Parent writeDirectory If direcotry is the current and Parent not Null Current = parent currentPath = cuurentPath without lastindexof // Then current.readDirecotry

Source Code	public void deleteDirectory()
	{
	clearDirSize();
	if (this.parent != null)
	{
	int index =
	this.parent.searchDirectory(new string(this.dir_name));
	if (index != -1)
	{
	this.parent.ReadDirectory();
	this.parent.entries.RemoveAt(index);
	this.parent.WriteDirectory();
	}
	}
	if (Program.current == this)
	{
	if (this.parent != null)
	{
	Program.current = this.parent;
	Program.currentPath =
	Program.currentPath.Substring(0,
	Program.currentPath.LastIndexOf('\\'));
	Program.current.ReadDirectory();
	}

	} FAT.writeFat(); }

Function 6	getMySizeOnDisk
Input	none
Output	Return Size
Processing	Get first_cluster from this.first_cluster Get next from getClusterNext(first_cluster) Then increase count by 1 (size) Then put clusterindex = next Then check if clusterindex != -1 Then get next of getClusterNext(clusterindex) And loop this until clusterindex == -1 Then return size after increase by 1 in every loop

Source Code	public int getMySizeOnDisk()
	{
	int size = 0;
	
	int clusterIndex = this.firs_cluster;
	
	int next = FAT.getClusterNext(clusterIndex);
	
	do
	{
	size++;
	clusterIndex = next;
	
	if (clusterIndex != -1)
	{
	next = FAT.getClusterNext(clusterIndex);
	}
	
	} while (clusterIndex != -1);
	
	return size;
	}

Function 7	canAddEntry
Input	DirecotryEntry d
Output	Boolean can add or not
Processing	Add 1 to entries count and multiple 32 to known needsize needCluster = needsize * 1024 Remindar of needsize if greater than 0 then increase needCluster++ neededClusters += d.dir_fileSize / 1024; If remindar of filesize greater than 0 increase needCluster

	if(getMySizeOnDisk()+FAT.getAvailableClusters()>neededClusters) If sizeonDisk and availableCluster greater than needCluster than can be true Finally return can

Source Code	public bool canAddEntry(DirectoryEntry d)
	{
	bool can = false;
	int needeSize = (entries.Count + 1) * 32;
	int neededClusters = needeSize * 1024;
	int rem = needeSize % 1024;
	if (rem > 0)
	neededClusters++;
	neededClusters += d.dir_fileSize / 1024;
	int rem2= d.dir_fileSize % 1024;
	
	if (rem2 > 0)
	neededClusters++;
	if(getMySizeOnDisk()+FAT.getAvailableClusters() > neededClusters)
	can = true;
	
	return can;
	}

Fourth Class: FILE: DirectoryEntry (inheritance)

Has
-public Directory parent -> refer to the parent of directory
-public string content -> refer content in file

Function 1	writeFile
Input	none
Output	void
Processing	Convert the content in byte in array called byteContent And split bytecontent in listofarrayofbyte If firt_Cluster not equal zero Do clusterFATIndex = this.firs_cluster; Else getemptycluster and put in clusterFatIndex Loop on listofarrayofbyte.count writeCluster of first index Then setnextofthisclusterindex = -1 if (lastCluster != -1) FAT.setClusterNext(lastCluster, clusterFATIndex); Then lastCluster = clusterindex Clusterindex = GetEmptyCulster();

Source Code	public void writeFile()
	{
	byte[] byteContent = ConvertContentToBytes(content);
	List<byte[]> listOfArrayOfBytes = VirtualDisk.splitBytes(byteContent);

```
            int clusterFATIndex;
            if (this.firs_cluster != 0)
            {
                clusterFATIndex = this.firs_cluster;
            }
            else
            {
                clusterFATIndex = FAT.GetEmptyCulster();
                this.firs_cluster = clusterFATIndex;
            }
            int lastCluster = -1;
            for (int i = 0; i < listOfArrayOfBytes.Count; i++)
            {
                if (clusterFATIndex != -1)
                {

VirtualDisk.writeCluster(clusterFATIndex,listOfArrayOfBytes[i])
;
                    FAT.setClusterNext(clusterFATIndex, -1);
                    if (lastCluster != -1)
                        FAT.setClusterNext(lastCluster,
clusterFATIndex);
                    lastCluster = clusterFATIndex;
                    clusterFATIndex = FAT.GetEmptyCulster();
                }
            }
        }
```

Function 2	ReadFile
Input	none
Output	void
Processing	Check firstCluster not equal zero Get culster from this.fir_cluster Get next from the getclusternext(cluster); And add range to ls (array of list of byte) of this cluster Get next of this cluster Untile next ==-1 Finally Convrtbytetocontent

Source Code	public void ReadFile()
	{
	if (this.firs_cluster != 0)
	{
	content = string.Empty;
	int cluster = this.firs_cluster;
	int next = FAT.getClusterNext(cluster);
	List<byte> ls = new List<byte>();
	do
	{
	
	ls.AddRange(VirtualDisk.readCluster(cluster));
	cluster = next;
	if (cluster != -1)
	next =
	FAT.getClusterNext(cluster);
	}
	while (next != -1);

	content = ConvertBytesToContent(ls.ToArray());
	
	//ASCIIEncoding encoding = new ASCIIEncoding();
	//content = encoding.GetString(ls.ToArray());
	
	}
	
	}

Function 3	deleteFile
Input	none
Output	void
Processing	Check firstCluster not equal zero And then cleardirSize(); If this parent of this directory not equal NUll Search name of current program in parent If found Parent.entries.remove(index) get from search Parent.writeDirectory(); Fat.writeFat();

Source Code	public void deleteFile()
	{
	if (this.firs_cluster != 0)
	{
	clearFileSize();
	}
	if (this.parent != null)
	{
	string dirName = new string(dir_name);
	int index = this.parent.searchDirectory(dirName);
	
	if(index != -1)
	{
	this.parent.entries.RemoveAt(index);
	this.parent.WriteDirectory();
	FAT.writeFat();
	}
	}
	}

Function 4	clearFileSize
Input	none
Output	void
Processing	Get culster from this.fir_cluster Get next from the getclusternext(cluster); Check clusterindex == 5 && next ==0 Not data found to clear Return ; Then setclusternext = (clusterindex,0) Clusterindex = next

	Check clusterindex not equal -1 Get Next Untile clusterindex == -1

Source Code	public void clearFileSize()
	{
	int clusterIndex = this.firs_cluster;
	
	int next = FAT.getClusterNext(clusterIndex);
	
	if (clusterIndex == 5 && next == 0)
	return;
	
	do
	{
	FAT.setClusterNext(clusterIndex, 0);
	clusterIndex = next;
	if (clusterIndex != -1)
	next =
	FAT.getClusterNext(clusterIndex);
	
	} while (clusterIndex != -1);
	
	}

The shell support the following internal commands:

Cd:

Function 4	moveTodir
Input	string p,bool usedInCD,bool isUsedInRD
Output	Object of Directory

Processing	Make object of Directory to be NULL
	Split p by (“\\”) and assign to arr
	If arr.length == 1 user put Name not Path
	If(arr[0] != ‘..’):
	Search Directory with this name in array
	If Not Found :
	Return -1 OR print this directory is not Found
	If Found:
	Get Name of Directory from current
	Then make Directory called D
	D. readDirectory
	Path = path.current of new Directory
	Path += “\\”+Name.trim()
	If(isUsedCD) => this true
	Program.currentPath = path;
	Else :
	Means the user want to go pervious page
	If the parent of current program not Null:
	D = program.current.parent;
	d.readDirectory;
	Path = program.currentPath without lastindex of “\\”
	if(usedInCD)
	Program.currentPath = path;
	Else: // means this floder with no Parent

	D = program.current;
	d.readDirectory();
	If User put Full instead of FileName:
	Make ListOfHandledPath list of String
	Then loop on arr.Length if index of this is not empty
	add in ListOfHandledPath
	Then make Root Directory and readDirectory
	If first index of ListOfHandledPath is Root(M: || m:)
	Loop on ListOfHandledPath.count
	{
	SearchDirectory(ListOfHandledPath[i])
	If found
	Make Directory of attrbuite
	Then add the name to Path
	}
	Program.currentPath = path;
	
	If we want to go back :
	ListOfHandledPath[0] == ".."
	
	If this parent not Equal Null
	D = d.parent
	E. readDirectory();
	Path = Program.currentPath;
	Then path go back and remove lastindex of “\\”
	And seek the program.currentPath = path

Source Code	public static void moveToDir(string path)
	{
	Directory dir = moveTodir(path,true,false);
	if (dir != null)
	{
	dir.ReadDirectory();
	Program.current = dir;
	}
	else
	{
	Console.WriteLine($" path {path} is not exists!");
	}
	}
	public static void moveToDirUsedInAnother(string path)
	{
	Directory dir = moveTodir(path, false, false);
	if (dir != null)
	{
	dir.ReadDirectory();
	Program.current = dir;
	}
	else
	{
	Console.WriteLine($" path {path} is not exists!");
	}
	}
	
	private static Directory moveTodir(string p,bool usedInCD,bool isUsedInRD)

```
        {
            Directory d = null;
            string[] arr = p.Split('\\');
            string path;
            if (arr.Length==1) // cd dirName
            {
                if (arr[0] != "..")
                {
                    int i =
Program.current.searchDirectory(arr[0]);
                    if (i == -1)
                        return null;//the directory is not
found
                    else
                    {
                        string nameOfDiserableFolder = new
string(Program.current.entries[i].dir_name); // we get
the name of the directory se seek to move to it
                        byte attr =
Program.current.entries[i].dir_attr;//also we get its
arrtributes
                        int fisrtcluster =
Program.current.entries[i].firs_cluster;
                        d = new
Directory(nameOfDiserableFolder, attr, fisrtcluster,
Program.current); //we take object of it to read its
content and to return it as a current path
                        d.ReadDirectory();
                        path = Program.currentPath; // we
take the current path to add to it the new directory
                        path += "\\" +
nameOfDiserableFolder.Trim();
                        if(usedInCD)
                            Program.currentPath =
path;//here we upadted the path M:>> -> m:/mohamed>>
                    }
                }
                else // .. means the user want to go to
the previous folder(parent)
                {
                    if (Program.current.parent != null)//
the current folder has a previous folder to back to it
                    {
                        d = Program.current.parent;
                        d.ReadDirectory();
                        path = Program.currentPath;
                        path = path.Substring(0,
path.LastIndexOf('\\')); // updating the current path
M:/mohamed -> M:
                        if(usedInCD)
                            Program.currentPath = path;
                    }
                    else // the current folder is the root
and there is no previous folder to go to it.
                    {
                        d = Program.current;
                        d.ReadDirectory();
                    }
                }
            }
            else if (arr.Length > 1)//the user enterd a
full path to go
            {
```

```
            List<string> ListOfHandledPath = new
List<string>();
            for (int i = 0; i < arr.Length; i++)
                if (arr[i] != "")
                    ListOfHandledPath.Add(arr[i]);

            Directory rootDirectory = new
Directory("M:", 0x10, 5, null);
            rootDirectory.ReadDirectory();


            if (ListOfHandledPath[0].Equals("m:") ||
ListOfHandledPath[0].Equals("M:")) // check if the root
folder the user entered is correct.
            {
                path = "M:";
                int howLongIsMyWay;
                if (isUsedInRD || usedInCD)
                {
                    howLongIsMyWay =
ListOfHandledPath.Count;
                }
                else {
                    howLongIsMyWay =
ListOfHandledPath.Count-1;
                }
                for (int i = 1; i < howLongIsMyWay;
i++) //ss -> mohamed sayed
                {
                    int j =
rootDirectory.searchDirectory(ListOfHandledPath[i]); //
serach for the next folder in the path
                    if (j != -1) // if found
                    {
                        Directory tempOfParent =
rootDirectory;
                        string newName = new
string(rootDirectory.entries[j].dir_name);// we get the
name of the directory se seek to move to it
                        byte attr =
rootDirectory.entries[j].dir_attr;//also we get its
arrtributes
                        int fc =
rootDirectory.entries[j].firs_cluster;
                        rootDirectory = new
Directory(newName, attr, fc, tempOfParent);
                        rootDirectory.ReadDirectory();
                        path += "\\" + newName.Trim(new
char[] { '\0', ' ' });
                    }
                    else//not found
                    {
                        return null;
                    }
                }
                d = rootDirectory;
                if(usedInCD)
                    Program.currentPath = path;
            }
            else if (ListOfHandledPath[0] ==
"..")//want to go back
            {
```

	d = Program.current;
	for (int i = 0; i < ListOfHandledPath.Count; i++)
	{
	if (d.parent != null)
	{
	d = d.parent;
	d.ReadDirectory();
	path = Program.currentPath;
	path = path.Substring(0, path.LastIndexOf('\\'));
	if(usedInCD)
	Program.currentPath = path;
	}
	else
	{
	break;
	}
	}
	}
	else
	return null;
	}
	return d;
	}

Dir: List the contents of directory

Function 1	dir
Input	none
Output	void
Processing	Loop on current entries count Check if this attr is folder or Direcotry If folder Console.WriteLine($"\t<DIR> {new string(Program.current.entries[i].dir_name)}"); Increase dCount++ If file Console.WriteLine($"\t{Program.current.entries[i].dir_fileSize} \t {new string(Program.current.entries[i].dir_name)}"); Increase fcount++; Then Print count of file and total file size Print count of directory with freespace By use 1024*1024 - int(Disk.length);

Source Code	public static void dir()
	{
	int fc = 0,dc = 0,fz_sum = 0;
	Console.WriteLine("Directory of " + Program.currentPath);

```
            Console.WriteLine();
            for (int i = 0; i < Program.current.entries.Count;
i++)
            {
                if (Program.current.entries[i].dir_attr == 0x0)
                {

Console.WriteLine($"\t{Program.current.entries[i].dir_fileSize}
\t {new string(Program.current.entries[i].dir_name)}");
                    fc++;
                    fz_sum +=
Program.current.entries[i].dir_fileSize;
                }
                else if (Program.current.entries[i].dir_attr ==
0x10)
                {
                    Console.WriteLine($"\t<DIR> {new
string(Program.current.entries[i].dir_name)}");
                    dc++;
                }
            }
            Console.WriteLine($"{"\t\t"}{fc} File(s)    {fz_sum}
bytes");
            Console.WriteLine($"{"\t\t"}{dc} Dir(s)
{VirtualDisk.getFreeSpace()} bytes free");
        }
```

Del: Deletes one or more files

Function 1	del
Input	`string fileName`
Output	void
Processing	Split fileName by \\ If path.length > 1 Do movetodirusedInAnother until last Index of Get FileName by get last index of array of path like that -> fileName = path[path.Length - 1]; Search on current program by FileName Check if file or Not by access attr If file Make object of this attribute And Delete File If Not File Print The System Cannot Find The file specified To return in root directory after moveit Directory rootDirectory = new Directory("M:", 0x10, 5, null); Program.current = rootDirectory; Program.current.ReadDirectory();

Rd: Removes a directory.

Function 1	rd
Input	`string name`

Output	void
Processing	Split Name by \\ Move to DIretory dir If dir not equal null Ask user for delete this folder or not If choise equal y Dir.deleteDirectory(); If dir null Print directory is not Exist

Source Code	public static void rd(string name)
	{
	
	string[] arr = name.Split('\\');
	Directory dir = moveTodir(name,false,true);
	if (dir != null)
	{
	Console.Write($"Are you sure that you want to delete {new string(dir.dir_name).Trim()} , please enter Y for yes or N for no:");
	string choice = Console.ReadLine().ToLower();
	if (choice.Equals("y"))
	dir.deleteDirectory();
	}
	else
	Console.WriteLine($"directory \" {arr[arr.Length-1]} \" is not exists!");
	
	}

type : Displays the contents of a text file.

Function 1	type
Input	string name
Output	void
Processing	Split fileName by \\ If path.length > 1 Do movetodirusedInAnother until last Index of Get FileName by get last index of array of path like that -> fileName = path[path.Length - 1]; Search on current program by FileName If found Make file object And readFile Then print Content If not found Print The System could not found the file specified To return in root directory after moveit Directory rootDirectory = new Directory("M:", 0x10, 5, null); Program.current = rootDirectory; Program.current.ReadDirectory();

Source Code	`public static void type(string name)`
	`{`
	`string[] path = name.Split("\\");`
	`if (path.Length > 1)`
	`{`
	`for (int i = 1; i < path.Length - 1; i++)`
	`moveToDirUsedInAnother(path[i]);`
	
	`name = path[path.Length - 1];`
	`}`
	
	`int j = Program.current.searchDirectory(name);`
	`if (j != -1)`
	`{`
	`int fc = Program.current.entries[j].firs_cluster;`
	`int sz = Program.current.entries[j].dir_fileSize;`
	`string content = null;`
	`FILE file = new FILE(name,0x0,fc,Program.current,content,sz);`
	`file.ReadFile();`
	`Console.WriteLine(file.content);`
	`}`
	`else {`
	`Console.WriteLine("The System could not found the file specified");`
	`}`
	`Directory rootDirectory = new Directory("M:", 0x10, 5, null);`
	`Program.current = rootDirectory;`
	`Program.current.ReadDirectory();`
	`}`

md : Creates a directory.

Function 1	makeFolder
Input	`string name`
Output	void
Processing	Split fileName by \\ If arr.length == 1 then he put folderName Search on current with no folder with the name user entered Cehck empty clusters (free space) to make a new folder Make DirectoryEntry and add in entries And WriteDirectory If this parent of current not equal null updatecontent of parent And writeDirectory and then Write Fat If not found of free space

	Print the disk is fully If arr.length > 1 // then user put full path Move to dir If dir equal null Print the directory is not exist If not null Check full disk Make object from directoryEntry Add this to entries Dir.writeDirectory Updatecontent of parent Partent writeDirctory

Source Code	`public static void makeFolder(string name)`
	`{`
	`string[] arr = name.Split('\\');`
	`if (arr.Length == 1) // md folderName`
	`{`
	`if (Program.current.searchDirectory(arr[0]) == -1)// there is no folder with the name user entered`
	`{`
	`if (FAT.GetEmptyCulster() != -1)//there is empty clusters (free space) to make a new folder`
	`{`
	`DirectoryEntry d = new DirectoryEntry(arr[0], 0x10, 0,0);`
	`Program.current.entries.Add(d);`
	`Program.current.WriteDirectory();`
	`if (Program.current.parent != null)`
	`{`
	`Program.current.parent.updateContent(Program.current.getDirectoryEntry());`
	`Program.current.parent.WriteDirectory();`
	`}`
	`FAT.writeFat();`
	`}`
	`else`
	`Console.WriteLine("The Disk is Full :(");`
	`}`
	`else`
	`Console.WriteLine($"{arr[0]} is aready existed :(");`
	`}`
	`else if (arr.Length > 1)`
	`{`
	`Directory dir = moveTodir(name,false,false);`
	`if (dir == null)`
	`Console.WriteLine($"The Path {name} Is not exist");`
	`else`
	`{`
	`if (FAT.GetEmptyCulster() != -1)//not full`
	`{`
	`DirectoryEntry d = new DirectoryEntry(arr[arr.Length - 1], 0x10, 0,0); //making the new`

	folder
	dir.entries.Add(d);// add it to the the folder we want to create into it a new folder
	dir.WriteDirectory();
	
	dir.parent.updateContent(dir.getDirectoryEntry());
	dir.parent.WriteDirectory();
	FAT.writeFat();
	}
	else
	Console.WriteLine("The Disk is Full :(");
	}
	}
	
	}

Rename: Renames a file

Function 1	rename
Input	string oldName, string newName
Output	void
Processing	search on directory by old Name If found Check search on directory by new_name If not found Make object of DirectoryEntry Check if file or Folder by object.attr To handle Name Then remove from entries(index that found in search) Add for entries insert(indext,object) writeDirctory If found Print Doublicate File Name exist or file cannot be found

Source Code	public static void rename(string oldName, string newName)
	{
	string[] path = oldName.Split("\\"); //old name could be path
	if (path.Length > 1)
	{
	for (int i = 1; i < path.Length - 1; i++)
	moveToDirUsedInAnother(path[i]);
	
	oldName = path[path.Length - 1];
	}
	
	int j = Program.current.searchDirectory(oldName);
	if (j != -1)
	{
	if (Program.current.searchDirectory(newName) == -1)
	{

	DirectoryEntry d = Program.current.entries[j];
	
	if (d.dir_attr == 0x0)
	{
	string[] fileName = newName.Split('.');
	char[] goodName = getProperFileName(fileName[0].ToCharArray(), fileName[1].ToCharArray());
	d.dir_name = goodName;
	}
	else if (d.dir_attr == 0x10)
	{
	char[] goodName = getProperDirName(newName.ToCharArray());
	d.dir_name = goodName;
	}
	
	Program.current.entries.RemoveAt(j);
	Program.current.entries.Insert(j, d);
	Program.current.WriteDirectory();
	}
	else
	{
	Console.WriteLine("Doublicate File Name exist or file cannot be found");
	}
	}
	else
	{
	Console.WriteLine("The System Cannot Find thr File specified");
	}
	
	Directory rootDirectory = new Directory("M:", 0x10, 5, null);
	Program.current = rootDirectory;
	Program.current.ReadDirectory();
	}

`import : import text file(s) from your computer`

Function 1	import
Input	`string dest`
Output	void
Processing	Check dest exist or Not If found realAllLine(dest) and assign to content Size = content.length Names = dest.Split("\\"); Name = names[names.legth-1] get last name of the path Search on this name in current If not found Make object of file And writeFile

	Make an object of DirectoryEntry to add this in current of program and write Directory If found Print file is already exist in your virtual disk

Source Code	`public static void import(string dest)`
	`{`
	`if (File.Exists(dest))`
	`{`
	`string content = File.ReadAllText(dest);`
	`int size = content.Length;`
	`string[] names = dest.Split("\\");`
	`string name = names[names.Length-1];`
	`int j = Program.current.searchDirectory(name);`
	`if (j == -1)`
	`{`
	`int fc;`
	`if (size > 0)`
	`{`
	`fc = FAT.GetEmptyCulster();`
	`}`
	`else`
	`{`
	`fc = 0;`
	`}`
	`FILE newFile = new FILE(name, 0X0, fc, Program.current, content, size);`
	
	`newFile.writeFile();`
	`//FAT.writeFat();`
	
	`DirectoryEntry d = new DirectoryEntry(new string(name), 0X0, fc, size);`
	`Program.current.entries.Add(d);`
	`Program.current.WriteDirectory();`
	`}`
	`else {`
	`Console.WriteLine($"{name} is already exist in your virtual disk");`
	`}`
	
	`}`
	`else {`
	`Console.WriteLine("The file you specified does not exist in your compuret");`
	`}`
	`}`

export : export text file(s) to your computer

Function 1	export
Input	`string source, string dest`
Output	void
Processing	Split source by \\ If path.length > 1 Do movetodirusedInAnother until last Index of

	Get source by get last index of array of path like that
	-> source = path[path.Length - 1];
	Search on current program by srouce
	If not found
	Check if Direcotry is Exist or Not
	If found
	Make file Object with content Null
	The
	streamWrite.write(content)
	Sw.flush(); // save the process of sw
	Sw.close(); // close the process of sw
	If not Found
	Print The system cannot find the path specified in the computer disk
	
	To return in root directory after moveit
	Directory rootDirectory = new Directory("M:", 0x10, 5, null);
	Program.current = rootDirectory;
	Program.current.ReadDirectory();

Source Code	public static void export(string source, string dest)
	{
	string[] path = source.Split("\\");
	if (path.Length > 1)
	{
	for (int i = 1; i < path.Length - 1; i++)
	moveToDirUsedInAnother(path[i]);
	
	source = path[path.Length - 1];
	}
	int j = Program.current.searchDirectory(source);
	if (j != -1)
	{
	if (System.IO.Directory.Exists(dest))
	{
	int fc = Program.current.entries[j].firs_cluster;
	int sz = Program.current.entries[j].dir_fileSize;
	string content = null;
	FILE file = new FILE(source, 0x0, fc, Program.current, content, sz);
	file.ReadFile();
	StreamWriter sw = new StreamWriter(dest + "\\" + source);
	sw.Write(file.content);
	sw.Flush();
	sw.Close();
	}
	else
	{
	Console.WriteLine("The system cannot find the path specified in the coputer disk");
	}
	
	}

	else
	{
	Console.WriteLine("The system cannot find the file you want to export in the virtual disk");
	}
	Directory rootDirectory = new Directory("M:", 0x10, 5, null);
	Program.current = rootDirectory;
	Program.current.ReadDirectory();
	}

Copy : Copies one or more files to another location

Function 1	copy
Input	string source, string dest
Output	void

Processing	SearchDirectory with Source Name in current Program
	
	If Found
	Myway = split dest by “\\”
	Loop on myway{
	moveToDIr(myway[i])
	}
	
	Now we seek in current program in Dest
	Then
	Search on Source in Dest Current
	If found
	Ask You Do you want to overwrite it ?, please enter Y for yes or N for no:
	
	if(y)
	Make object D of DirectoryEntry
	Add in program.current.entries
	If not Found
	Make object D of DirectoryEntry
	Add in program.current.etntries
	Program.current.writeDirectory();
	
	Then to return seek in root Directory
	
	Make Object Root Directory
	Root.writeDirectory
	Program.current = root
	
	If FIle Not Found
	Print the File Is Not Existed

Copy	public static void copy(string source, string dest)
	{
	int j = Program.current.searchDirectory(source);
	int fc = Program.current.entries[j].firs_cluster;
	int sz = Program.current.entries[j].dir_fileSize;

```
            if (j != -1)
            {
                string[] myWay = dest.Split("\\");
                for (int i = 1; i < myWay.Length; i++)
                    moveToDirUsedInAnother(myWay[i]);


                int x =
Program.current.searchDirectory(source);
                if (x != -1)
                {
                    Console.Write("The File is aleary
existed, Do you want to overwrite it ?, please enter Y
for yes or N for no:");
                    string choice =
Console.ReadLine().ToLower();
                    if (choice.Equals("y"))
                    {


                        DirectoryEntry d = new
DirectoryEntry(new string(source), 0X0, fc, sz);

                        Program.current.entries.Add(d);

                    }
                    else
                    {
                        return;
                    }
                }
                else
                {


                    DirectoryEntry d = new
DirectoryEntry(new string(source), 0X0, fc, sz);

                    Program.current.entries.Add(d);

                    Program.current.WriteDirectory();

                }

                Directory rootDirectory = new
Directory("M:", 0x10, 5, null);
                Program.current = rootDirectory;
                Program.current.ReadDirectory();

            }
            else
            {
                Console.WriteLine($"The File ${source} Is
Not Existed");
            }
        }
```

help : Provides Help information for commands

Function 1	doHelp

Input	Struct of Token
Output	void
Processing	If token.value == null Print all help Command If token.value == value of Command in Shell Then print help of this only command If token.value == value unkown Then print defalut case Unkown argument

Source Code	public void doHelp(Token token)
	{
	if (token.value == null)
	{
	Console.WriteLine(help_command);
	return;
	}
	switch (token.value)
	{
	case "cd":
	Console.WriteLine(help_cd);
	Console.WriteLine("The Syantax :\ncd [Path]\n");
	break;
	case "dir":
	Console.WriteLine(help_dir);
	Console.WriteLine("The Syntax :\nDIR [d:][path][filename] [/A:(attributes)] [/O:(order)] [/B][/C][/CH][/L][/S][/P] [/W]\n");
	break;
	case "cls":
	Console.WriteLine(help_cls);
	Console.WriteLine("The Syntax :\ncls\n");
	break;
	case "quit":
	Console.WriteLine(help_quit);
	Console.WriteLine("The Syntax :\nquit\n");
	break;
	case "copy":
	Console.WriteLine(help_copy);
	Console.WriteLine("The Syntax :\ncopy [/d] [/v] [/n] [/y | /-y] [/z] [/l] [/a | /b] source [/a | /b] [+ source [/a | /b] [+ ...]] [destination [/a | /b]] [/?]\n");
	break;
	case "del":
	Console.WriteLine(help_del);
	Console.WriteLine("The Syntax :\ndel [/p] [/f] [/s] [/q] [/a[:]] filename [/?]\n");
	break;
	case "help":
	Console.WriteLine(help_help);
	Console.WriteLine("The Syntax :\nor help [command]\n");
	break;
	case "md":
	Console.WriteLine(help_md);
	Console.WriteLine("The Syntax :\nmd [<drive>:]<path>\n");

```
                    break;
                case "rd":
                    Console.WriteLine(help_rd);
                    Console.WriteLine("The Syntax :\nRD
[/S] [/Q] [drive:]path\n");
                    break;
                case "rename":
                    Console.WriteLine(help_rename);
                    Console.WriteLine("The
Syntax :\nRENAME [drive:][path]filename1
filename2.\n");
                    break;
                case "type":
                    Console.WriteLine(help_type);
                    Console.WriteLine("The Syntax :\nTYPE
[drive:][path]filename\n");
                    break;
                default:
                    Console.WriteLine("Unkown argument " +
token.value + " ..");
                    break;
            }
        }
```

Thank You