# MINI_FAT_FILE_SYSTEM

Wagdy yousry mohammad ➔ sec 3 is

Ahmed Abdalraouf Abdalnasser ➔ sec 1 is

First Class: virtual_disk

| Function 1 | `Prepare_Fat_Table` |
|---|---|
| Input | string path |
| Output | |
| Processing | Check if file of fat exist or not<br>If not found<br>Create file and write file cluster (1024) bits<br>And prepare Fat (array of intger ) size [1024]<br>Make ROOT directory and make this the current program<br>Then write in Fat |
| Source Code | `public void Prepare_Fat_Table()`<br>`{`<br>`string path = Directory.GetCurrentDirectory() + @"\\Virtual_disk.txt";`<br>`FileInfo Virtual_disk_txt = new FileInfo(path);`<br>`directory root = new directory("H", 1, 5, 0, null);`<br>`MINI_FAT.setNext(5, -1);`<br>`Program.c_dic = root;` |

```
            if (File.Exists(path))
            {
                MINI_FAT.fatTabel = MINI_FAT.Read_fat();

                root.Read_directory();
            }
            else
            {

                FileStream wt = 
Virtual_disk_txt.Open(FileMode.Create, 
FileAccess.ReadWrite);
                for (int i = 0; i < 1024; i++)
                {
                    wt.WriteByte(0);
                }
                for (int i = 0; i < 4 * 1024; i++)
                {
                    wt.WriteByte((byte)'*');
                }
                for (int i = 0; i < 1019 * 1024; i++)
                {
                    wt.WriteByte((byte)'#');
                }
                wt.Close();
                MINI_FAT.Prepare_Fat_Table();
                root.write_directory();
                MINI_FAT.write_fat();
            }
        }
```

| Function 2 | Write_block |
|---|---|
| Input | int clusterIndex_ byte[] bytes |
| Output | Void |
| Processing | Put seek begin on clusterIndex * 1024<br>And write an of byte from 0 to length of array |
| Source Code | |

```
public static void write_block(byte[] data, int index)
        {

            string path = Directory.GetCurrentDirectory() + 
@"\\Virtual_disk.txt";

            FileStream Virtual_disk_text = new 
FileStream(path, FileMode.Open, FileAccess.ReadWrite);
            Virtual_disk_text.Seek(1024 * index, 
SeekOrigin.Begin);
            Virtual_disk_text.Write(data, 0, data.Length);
            Virtual_disk_text.Close();

        }
```

| Function 3 | Read_block |
|---|---|
| Input | int clusterIndex |
| Output | byte[] bytes |
| Processing | Put seek begin on clusterIndex * 1024<br>And readan of byte from 0 to length of array |

| | |
|---|---|
| Source Code | public static byte[] read_block(int index)<br>        {<br>            string path = Directory.GetCurrentDirectory() + @"\\Virtual_disk.txt";<br>            FileStream Virtual_disk_text = new FileStream(path, FileMode.Open, FileAccess.ReadWrite);<br>            Virtual_disk_text.Seek(1024 * index, SeekOrigin.Begin);<br>            Byte[] bt = new Byte[1024];<br>            Virtual_disk_text.Read(bt, 0, bt.Length);<br>            Virtual_disk_text.Close();<br>            return bt;<br>        } |

Second Class: FAT

| | |
|---|---|
| Function 1 | Prepare_Fat_Table() |
| Input | None |
| Output | Void |
| Processing | Loop for length of array and put:<br>index 0 and 4 value 0<br>index between 0 and 4 value index + 1<br>Else put value of them 0 |
| Source Code | public static void Prepare_Fat_Table()<br>        {<br>            fatTabel[0] = -1;<br>            fatTabel[1] = 2;<br>            fatTabel[2] = 3;<br>            fatTabel[3] = 4;<br>            fatTabel[4] = -1;<br><br><br>        } |

| Function 2 | write_fat() |
|---|---|
| Input | None |
| Output | Void |
| Processing | Make array of byte length equal fat.length * 4<br>And convert array of intger to array of byte<br>then split array of byte clusters 1024 (array) and put all of them in list of array<br>Then loop on count of this list and write cluster all of them in Disk File |
| Source Code | public static void write_fat()<br>{<br>string path = Directory.GetCurrentDirectory() + @"\\Virtual_disk.txt";<br>FileStream wt = new FileStream(path, FileMode.Open, FileAccess.ReadWrite);<br>wt.Seek(1024, SeekOrigin.Begin);<br>Byte[] bt = new Byte[1024 * 4];<br>Buffer.BlockCopy(fatTabel, 0, bt, 0, bt.Length);<br>wt.Write(bt, 0, bt.Length);<br>wt.Close();<br>} |

| Function 3 | Read_fat() |
|---|---|
| Input | None |
| Output | Void |
| Processing | Make buffer (array of byte ) size (4096) first 4 cluster<br>And loop of them cluster and put of them in list of array of byte<br>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 int[] Read_fat()<br>{<br>string path = Directory.GetCurrentDirectory() + @"\\Virtual_disk.txt";<br>FileStream rd = new FileStream(path, FileMode.Open, FileAccess.ReadWrite);<br>rd.Seek(1024, SeekOrigin.Begin);<br>Byte[] bt = new Byte[1024 * 4];<br>rd.Read(bt, 0, bt.Length);<br>rd.Close();<br>Buffer.BlockCopy(bt, 0, fatTabel, 0, fatTabel.Length);<br>return fatTabel;<br>} |

| Function 4 | getAvaliablIndex() |
|---|---|
| Input | None |

| | |
|---|---|
| Output | Index of empty cluster |
| Processing | Loop for fat length<br>And check if index of them value equal 0<br>Return the index<br>Else return -1 |
| Source Code | `public static int getAvaliablIndex()`<br>`        {`<br>`            int[] S = Read_fat();`<br>`            for (int i = 0; i < 1024; i++)`<br>`            {`<br>`                if (S[i] == 0)`<br>`                {`<br>`                    return i;`<br>`                }`<br><br>`            }`<br>`            return -1;`<br>`        }` |

| | |
|---|---|
| Function 4 | `getAvaliableBlock()` |
| Input | None |
| Output | Count of Availabele Clusters |
| Processing | Loop for fat length<br>And check if index of them valued equal 0<br>Increase count++<br>After loop<br>Return Count |
| Source Code | `public static int getAvaliableBlock()`<br>`        {`<br>`            int[] S = Read_fat();`<br>`            int no = 0;`<br>`            for (int i = 0; i < 1024; i++)`<br>`            {`<br>`                if (S[i] == 0)`<br>`                {`<br>`                    no++;`<br>`                }`<br><br>`            }`<br>`            return (no == 0 ? -1 : no);`<br>`        }` |

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<br>Then substring 11 char of string name<br>Then for loop in all entries<br>Get name of all entries and equal with string name |

| | |
|---|---|
| | If found return index of this entry<br>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<br>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<br>Make list of entries<br>This add range in list of array of byte read cluster until next equal -1 then out of this loop<br>After that<br>Make loop of ls count<br>And nake array of byte size 32 byte<br>And assign from ls to small array<br>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
```

|  |  |
|---|---|
|  | `{`<br>`    ls.AddRange(VirtualDisk.readCluster(cluster));`<br>`    cluster = next;`<br>`    if (cluster != -1)`<br>`        next = FAT.getClusterNext(cluster);`<br>`}`<br>`while (next != -1);`<br>`for (int i = 0; i < ls.Count; i++)`<br>`{`<br>`    byte[] b = new byte[32];`<br>`    for (int k = i * 32, m = 0; m < b.Length && k < ls.Count;`<br>`m++, k++)`<br>`    {`<br>`        b[m] = ls[k];`<br>`    }`<br>`    if (b[0] == 0)`<br>`        break;`<br>`    entries.Add(makeDirectory(b));`<br>`}`<br>`}`<br>`}` |

| Function 4 | WriteDirectory |
|---|---|
| Input | none |
| Output | void |
| Processing | Loop on entries count<br>And take index of them and convert in byte (32 byte)<br>And assign them take in account offest in big array of byte called DirorFilesinBytes<br>After that split big array in list of array byte 1024 called bytesls<br>Check first_cluster not equal zero<br>Put clusterindex the value of fisrt_cluster<br>If not put clusterindex value FAT.GetEmptyCulster();<br>Loop in bytesls<br>Check fully disk or not<br>Then write cluster of all index of list<br>Put clusternext equal -1 in this index<br>Then check   if (lastCluster != -1)<br>FAT.setClusterNext(lastCluster,clusterFATIndex);<br>First lastCluster equal -1 by defalut then not access this if<br><br>lastCluster = clusterFATIndex;<br>Then put lastCluster value of index of cluster<br>clusterFATIndex = FAT.GetEmptyCulster();<br>And new of cluster index get empty cluster in Fat<br><br>If entries count eqaul zero then no data found<br>To write then put first cluster eqaul 0<br><br>If this directory has parent (not NULL)<br>Update content of parent<br>Then call fun writedirecotry again<br>And finally writeFat |
| Source Code | `public void WriteDirectory()`<br>`{`<br>`    byte[] dirsorfilesBYTES = new byte[entries.Count * 32];`<br>`    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;<br>Check if the parent of this direcotry not null<br>Search on dir_name of program current<br>If found<br>Parent ReadDirectory<br>Remove from parent entries this index<br>Parent writeDirectory<br><br>If direcotry is the current and Parent not Null<br>Current = parent<br>currentPath = cuurentPath without lastindexof //<br>Then current.readDirecotry |

<table>
<tr><td></td><td></td></tr>
<tr><td>Source Code</td><td>

```
        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();
        }
```

</td></tr>
</table>

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

```
        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);
                }
```

</td></tr>
</table>

| | |
|---|---|
| | } while (clusterIndex != -1);<br>return size;<br>} |

| | |
|---|---|
| Function 7 | canAddEntry |
| Input | DirecotryEntry d |
| Output | Boolean can add or not |
| Processing | Add 1 to entries count and multiple 32 to known needsize<br>needCluster = needsize * 1024<br>Remindar of needsize if greater than 0 then increase needCluster++<br>neededClusters += d.dir_fileSize / 1024;<br>If remindar of filesize greater than 0 increase needCluster<br>if(getMySizeOnDisk()+FAT.getAvailableClusters()>neededClusters)<br>If sizeonDisk and availableCluster greater than needCluster than can be true<br>Finally return can |
| Source Code | public bool canAddEntry(DirectoryEntry d)<br>{<br>bool can = false;<br>int needeSize = (entries.Count + 1) * 32;<br>int neededClusters = needeSize * 1024;<br>int rem = needeSize % 1024;<br>if (rem > 0)<br>neededClusters++;<br>neededClusters += d.dir_fileSize / 1024;<br>int rem2= d.dir_fileSize % 1024;<br>if (rem2 > 0)<br>neededClusters++;<br>if(getMySizeOnDisk()+FAT.getAvailableClusters() > neededClusters)<br>can = true;<br>return can;<br>} |

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<br>And split bytecontent in listofarrayofbyte |

| | |
|---|---|
| | If firt_Cluster not equal zero<br>Do clusterFATIndex = this.firs_cluster;<br>Else getemptycluster and put in clusterFatIndex<br><br>Loop on listofarrayofbyte.count<br>writeCluster of first index<br>Then setnextofthisclusterindex = -1<br>if (lastCluster != -1)<br>FAT.setClusterNext(lastCluster, clusterFATIndex);<br>Then lastCluster = clusterindex<br>Clusterindex = GetEmptyCulster(); |
| Source Code | public void writeFile()<br>{<br>byte[] byteContent = ConvertContentToBytes(content);<br>List<byte[]> listOfArrayOfBytes = VirtualDisk.splitBytes(byteContent);<br><br>int clusterFATIndex;<br>if (this.firs_cluster != 0)<br>{<br>clusterFATIndex = this.firs_cluster;<br>}<br>else<br>{<br>clusterFATIndex = FAT.GetEmptyCulster();<br>this.firs_cluster = clusterFATIndex;<br>}<br>int lastCluster = -1;<br>for (int i = 0; i < listOfArrayOfBytes.Count; i++)<br>{<br>if (clusterFATIndex != -1)<br>{<br>VirtualDisk.writeCluster(clusterFATIndex,listOfArrayOfBytes[i]);<br>FAT.setClusterNext(clusterFATIndex, -1);<br>if (lastCluster != -1)<br>FAT.setClusterNext(lastCluster, clusterFATIndex);<br>lastCluster = clusterFATIndex;<br>clusterFATIndex = FAT.GetEmptyCulster();<br>}<br>}<br>} |

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

<table>
<tr><td>Source Code</td><td>

```
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());

    }

}
```

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

```
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();
        }
    }
}
```

</td></tr>
</table>

| | |
|---|---|
| Function 4 | clearFileSize |

| | |
|---|---|
| Input | none |
| Output | void |
| Processing | Get culster from this.fir_cluster<br>Get next from the getclusternext(cluster);<br>Check clusterindex == 5 && next ==0<br>Not data found to clear<br>Return ;<br><br>Then setclusternext = (clusterindex,0)<br>Clusterindex = next<br>Check clusterindex not equal -1<br>Get Next<br>Untile clusterindex == -1 |
| Source Code | public void clearFileSize()<br>{<br>int clusterIndex = this.firs_cluster;<br><br>int next = FAT.getClusterNext(clusterIndex);<br><br>if (clusterIndex == 5 && next == 0)<br>return;<br><br>do<br>{<br>FAT.setClusterNext(clusterIndex, 0);<br>clusterIndex = next;<br>if (clusterIndex != -1)<br>next = FAT.getClusterNext(clusterIndex);<br><br>} while (clusterIndex != -1);<br><br>} |

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<br>Split p by (“\\”) and assign to arr<br>If arr.length == 1 user put Name not Path<br>If(arr[0] != ‘..’):<br>Search Directory with this name in array<br>If Not Found :<br>Return -1 OR print this directory is not Found<br>If Found:<br>Get Name of Directory from current<br>Then make Directory called D<br>D. readDirectory<br>Path = path.current of new Directory<br>Path += “\\”+Name.trim()<br>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<br>{<br>break;<br>}<br>}<br>}<br>else<br>return null;<br>}<br>return d;<br>} |

## Dir: List the contents of directory

| | |
|---|---|
| Function 1 | dir |
| Input | none |
| Output | void |
| Processing | Loop on current entries count<br>Check if this attr is folder or Direcotry<br>If folder<br>Console.WriteLine($"\t<DIR> {new string(Program.current.entries[i].dir_name)}");<br>Increase dCount++<br>If file Console.WriteLine($"\t{Program.current.entries[i].dir_fileSize} \t {new string(Program.current.entries[i].dir_name)}");<br>Increase fcount++;<br><br>Then<br>Print count of file and total file size<br>Print count of directory with freespace<br>By use 1024*1024 - int(Disk.length); |
| Source Code | public static void dir()<br>{<br>int fc = 0,dc = 0,fz_sum = 0;<br>Console.WriteLine("Directory of " + Program.currentPath);<br>Console.WriteLine();<br>for (int i = 0; i < Program.current.entries.Count; i++)<br>{<br>if (Program.current.entries[i].dir_attr == 0x0)<br>{<br><br>Console.WriteLine($"\t{Program.current.entries[i].dir_fileSize} \t {new string(Program.current.entries[i].dir_name)}");<br>fc++;<br>fz_sum += Program.current.entries[i].dir_fileSize;<br>}<br>else if (Program.current.entries[i].dir_attr == 0x10)<br>{<br>Console.WriteLine($"\t<DIR> {new string(Program.current.entries[i].dir_name)}");<br>dc++;<br>} |

| | |
|---|---|
| | }<br>Console.WriteLine($"{"\t\t"}{fc} File(s)      {fz_sum} bytes");<br>Console.WriteLine($"{"\t\t"}{dc} Dir(s)      {VirtualDisk.getFreeSpace()} bytes free");<br>} |

Del: Deletes one or more files

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

Rd: Removes a directory.

| | |
|---|---|
| Function 1 | rd |
| Input | string name |
| Output | void |
| Processing | Split Name by \\<br>Move to DIretory dir<br>If dir not equal null<br>Ask user for delete this folder or not<br>If choise equal y<br>Dir.deleteDirectory();<br>If dir null<br>Print directory is not Exist |
| Source Code | public static void rd(string name)<br>{<br><br><br>string[] arr = name.Split('\\');<br>Directory dir = moveTodir(name,false,true);<br>if (dir != null)<br>{<br>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:");<br>string choice = Console.ReadLine().ToLower(); |

<table>
<tr><td></td><td>

```
                    if (choice.Equals("y"))
                        dir.deleteDirectory();
                }
                else
                    Console.WriteLine($"directory \" {arr[arr.Length-1]} \" is not exists!");

            }
```

</td></tr>
</table>

type : Displays the contents of a text file.

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

```
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");
    }
```

</td></tr>
</table>

| | |
|---|---|
| | Directory rootDirectory = new Directory("M:", 0x10, 5, null);<br>Program.current = rootDirectory;<br>Program.current.ReadDirectory();<br>} |

### md : Creates a directory.

| | |
|---|---|
| Function 1 | makeFolder |
| Input | string name |
| Output | void |
| Processing | Split fileName by \\<br>If arr.length == 1 then he put folderName<br>Search on current with no folder with the name user entered<br><br>Cehck empty clusters (free space) to make a new folder<br><br>Make DirectoryEntry and add in entries<br>And WriteDirectory<br><br>If this parent of current not equal null<br>updatecontent of parent<br>And writeDirectory and then<br>Write Fat<br><br>If not found of free space<br>Print the disk is fully<br><br><br>If arr.length > 1 // then user put full path<br>Move to dir<br>If dir equal null<br>Print the directory is not exist<br>If not null<br>Check full disk<br>Make object from directoryEntry<br>Add this to entries<br>Dir.writeDirectory<br>Updatecontent of parent<br>Partent writeDirctory |
| Source Code | public static void makeFolder(string name)<br>{<br>string[] arr = name.Split('\\');<br>if (arr.Length == 1) // md folderName<br>{<br>if (Program.current.searchDirectory(arr[0]) == -1)// there is no folder with the name user entered<br>{<br>if (FAT.GetEmptyCulster() != -1)//there is empty clusters (free space) to make a new folder<br>{<br>DirectoryEntry d = new DirectoryEntry(arr[0], 0x10, 0,0);<br>Program.current.entries.Add(d);<br>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<br>If found<br>Check search on directory by new_name<br>If not found<br>Make object of DirectoryEntry<br>Check if file or Folder by object.attr<br>To handle Name<br>Then remove from entries(index that found in search)<br>Add for entries insert(indext,object)<br>writeDirctory<br>If found<br>Print<br>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<br>If found<br>realAllLine(dest) and assign to content<br>Size = content.length<br>Names = dest.Split("\\");<br>Name = names[names.legth-1] get last name of the path<br>Search on this name in current<br>If not found<br>Make object of file<br>And writeFile<br>Make an object of DirectoryEntry to add this in current of program and write Directory<br><br>If found<br>Print file is already exist in your virtual disk |
| Source Code | public static void import(string dest)<br>{<br>if (File.Exists(dest))<br>{<br>string content = File.ReadAllText(dest);<br>int size = content.Length;<br>string[] names = dest.Split("\\");<br>string name = names[names.Length-1];<br>int j = Program.current.searchDirectory(name);<br>if (j == -1)<br>{<br>int fc;<br>if (size > 0)<br>{<br>fc = FAT.GetEmptyCulster();<br>}<br>else<br>{<br>fc = 0;<br>}<br>FILE newFile = new FILE(name, 0X0, fc, Program.current, content, size);<br><br>newFile.writeFile();<br>//FAT.writeFat();<br><br>DirectoryEntry d = new DirectoryEntry(new string(name), 0X0, fc, size);<br>Program.current.entries.Add(d);<br>Program.current.WriteDirectory();<br>}<br>else {<br>Console.WriteLine($"{name} is already exist in your virtual disk");<br>}<br><br>}<br>else {<br>Console.WriteLine("The file you specified does not exist in your compuret");<br>}<br>} |

| | |
|---|---|

## export : export text file(s) to your computer

| Function 1 | export |
|---|---|
| Input | string source, string dest |
| Output | void |
| Processing | Split source by \\<br>If path.length > 1<br>Do movetodirusedInAnother until last Index of<br>Get source by get last index of array of path like that -> source = path[path.Length - 1];<br>Search on current program by srouce<br>If not found<br>Check if Direcotry is Exist or Not<br>If found<br>Make file Object with content Null<br>The<br>streamWrite.write(content)<br>Sw.flush(); // save the process of sw<br>Sw.close(); // close the process of sw<br>If not Found<br>Print The system cannot find the path specified in the computer disk<br><br>To return in root directory after moveit<br>Directory rootDirectory = new Directory("M:", 0x10, 5, null);<br>Program.current = rootDirectory;<br>Program.current.ReadDirectory(); |
| Source Code | public static void export(string source, string dest)<br>{<br>string[] path = source.Split("\\");<br>if (path.Length > 1)<br>{<br>for (int i = 1; i < path.Length - 1; i++)<br>moveToDirUsedInAnother(path[i]);<br><br>source = path[path.Length - 1];<br>}<br>int j = Program.current.searchDirectory(source);<br>if (j != -1)<br>{<br>if (System.IO.Directory.Exists(dest))<br>{<br>int fc = Program.current.entries[j].firs_cluster;<br>int sz = Program.current.entries[j].dir_fileSize;<br>string content = null;<br>FILE file = new FILE(source, 0x0, fc, Program.current, content, sz);<br>file.ReadFile();<br>StreamWriter sw = new StreamWriter(dest + "\\" + source);<br>sw.Write(file.content);<br>sw.Flush();<br>sw.Close();<br>}<br>else<br>{ |

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

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<br><br>If Found<br>Myway = split dest by “\\”<br>Loop on myway{<br>moveToDIr(myway[i])<br>}<br><br>Now we seek in current program in Dest<br>Then<br>Search on Source in Dest Current<br>If found<br>Ask You   Do you want to overwrite it ?, please enter Y for yes or N for no:<br><br>if(y)<br>Make object D of DirectoryEntry<br>Add in program.current.entries<br>If not Found<br>Make object D of DirectoryEntry<br>Add in program.current.etntries<br>Program.current.writeDirectory();<br><br>Then to return seek in root Directory<br><br>Make Object Root Directory<br>Root.writeDirectory<br>Program.current = root<br><br>If FIle Not Found<br>Print the File Is Not Existed |
| Copy | public static void copy(string source, string dest)<br>{<br>int j = Program.current.searchDirectory(source);<br>int fc = Program.current.entries[j].firs_cluster;<br>int sz = Program.current.entries[j].dir_fileSize;<br><br>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<br>Print all help Command<br>If token.value == value of Command in Shell<br>Then print help of this only command |

| | |
|---|---|
| | If token.value == value unkown<br>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");<br>break;<br>default:<br>Console.WriteLine("Unkown argument " + token.value + " ..");<br>break;<br>}<br>} |