CSC 415: Operating System Principles

File System

Team name: Fantastic Four

Members:
Justin Adriano 921719692
Anudeep Katukojwala 922404701
Kristian Goranov 922003001
Pranjal Newalkar 922892162

Github: pranjal2410

Project Link - GitHub

Hex dump (after initializing our structure) :

VCB: spans one block.
Magic number is 72 75 6F 46 74 6E 61 46.
Volume Size is 10,000,000 and is 0x989680 in hex.
Block size is 512 and is 0x200
FreeSpace location is 1and is 0x01.
Root directory location is 6 and in hex 0x06.

FreeSpace:
Spans five blocks and is highlighted in yellow.
The VCB, the FreeSpace bitMap, and root directory are indicated by 0xFF = 1111111 for VCB taking up block 0, bitmap blocks 1-5, root the root directory taking up six blocks.

Root Directory:
Location: start of the root directory is block 6 for both ".", and "..". The root directory spans 6 blocks with a total of 10 Directory entries for now.
Size of root directory in two spots for ".", and ".." is 2880 or 0xB40.
Filename goes on for 256 bytes and 0x2E is ASCII for ".".
TimeStamp: for created and last modified both the same:
000F10: 52 59 5B 63 00 00 00 00 52 59 5B 63 00 00 00 00 |

And repeats after this line for the ".." directory.
We did not initialize any of the directory entries yet. But we will have a way of knowing they are not initialized.

Dumping file SampleVolume, starting at block 1 for 11 blocks:

```
000200: 72 75 6F 46 74 6E 61 46 80 96 98 00 00 02 00 00 | ruoFtnaF���.....
```

```
000210: 01 00 00 00 06 00 00 00  00 00 00 00 00 00 00 00 | ................
000220: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000230: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000240: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000250: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000260: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000270: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000280: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000290: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0002F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000300: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000310: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000320: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000330: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000340: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000350: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000360: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000370: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000380: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000390: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0003F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000400: FF F0 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ��..............
000410: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000420: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000430: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000440: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000450: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000460: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

```
000470: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000480: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000490: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0004F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000500: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000510: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000520: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000530: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000540: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000550: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000560: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000570: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000580: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000590: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0005F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000600: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000610: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000620: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000630: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000640: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000650: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000660: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000670: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000680: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000690: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

```
0006D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000700: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000710: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000720: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000730: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000740: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000750: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000760: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000770: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000780: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000790: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000800: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000810: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000820: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000830: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000840: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000850: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000860: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000870: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000880: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000890: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0008F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000900: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000910: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

```
000920: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000930: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000940: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000950: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000960: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000970: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000980: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000990: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0009F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000A00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A20: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000A90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000AF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000B00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B20: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

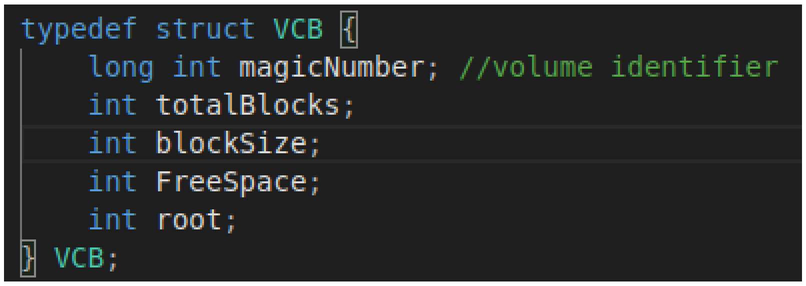
```
000B80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000B90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000BF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000C00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C20: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000C90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000CF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000D00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D20: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000D90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000DA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000DB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000DC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000DD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

```
000DE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000DF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000E00: 06 00 00 00 40 0B 00 00  06 00 00 00 2E 00 00 00 | ....@...........
000E10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E20: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000E90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000EA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000EB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000EC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000ED0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000EE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000EF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000F00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F10: 52 59 5B 63 00 00 00 00  52 59 5B 63 00 00 00 00 | RY[c....RY[c....
000F20: 06 00 00 00 40 0B 00 00  06 00 00 00 2E 2E 00 00 | ....@...........
000F30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F80: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000F90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000FF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

001000: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001010: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001020: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

```
001030: 52 59 5B 63 00 00 00 00  52 59 5B 63 00 00 00 00 | RY[c....RY[c....
001040: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001050: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001060: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001070: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001080: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001090: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0010F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

001100: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001110: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001120: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001130: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001140: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001150: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001160: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001170: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001180: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001190: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0011F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

001200: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001210: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001220: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001230: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001240: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001250: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001260: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001270: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001280: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

Description / what we did:

Description of the VCB structure:



```
typedef struct VCB {
    long int magicNumber; //volume identifier
    int totalBlocks;
    int blockSize;
    int FreeSpace;
    int root;
} VCB;
```

We chose to go with a bitMap for our FreeSpace management so we only needed to store the location of where it begins. How far it spans can be calculated with totalBlocks and blockSize.
And similarly we only needed a location for our root directory.

Description of the Free Space structure:

We first initialize the free space by calling a function we create called initFreeSpace. We allocated the size of five blocks to the bitmap pointer and initialized the bits in the freespace. To mark occupied blocks we used 1's and 0's for free space. We set bits 1-5 in the bitmap occupied for the bitmap itself so the initial freespace is 11111100. The first bit of our bitmap is our volume control block.

For free space management, we have used bitmap, where each bit indicates a block.
We have defined 0 as a free block and 1 as an allocated block in our bitmap. Our free space allocation algorithm works the following way: We take size (in bytes) as our parameter. This size will be given to our function by the user or any function in our file system that needs free space.
Once our function is called, based on the size given as input, we calculate the number of blocks that are needed as free space. We have defined 512 bytes as the size for each block. Now, we iterate through the bitmap array the following way: To find free

blocks, we take each element in the bitmap array one by one from the 0th index, check each bit in that element (or byte) and find the contiguous free blocks which would satisfy the request. Once we find enough contiguous free blocks, we return the block number from which the contiguous free blocks start. Our free space algorithm searches the bitmap in a linear way, takes each byte and checks each bit to find the free space blocks.

===

Free Space Management Functions:

Helper functions to manage free space:

int isTheBitFree(unsigned char input, int bitLocationInByte);

This function is used to know if the bit we specified in a byte is free or not.
This function needs a char value which is of 1 byte, and an int value from the range of 0 to 7, and returns zero or one based on the parameters.
Based on the bit location in the byte, we have a hex value that we copy into a temporary variable. We then bitwise AND that hex value with our input parameter and return 0 or 1 based on the output of bitwise AND

unsigned char hexValueOfClearBit(int offset);

A simple helper routine to get the hex value of the offset, which is used in freeTheBlocks function to clear the bits.
Returns an unsigned char.
For each bit location in a byte (from 0 to 7), we have defined a hex value, and we use a switch statement to get the required hex value based on the offset value in the parameter and return that hex value.

unsigned char hexValueOfSetBit(int offset);

A simple helper routine to get the hex value of the offset, which is used in setTheBlocks function to set the bits.
Returns an unsigned char.

For each bit location in a byte (from 0 to 7), we have defined a hex value, and we use a switch statement to get the required hex value based on the offset value in the parameter and return that hex value.

<u>void freeTheBlocks(int startBlock, int numberOfBlocks);</u>

This function takes the starting block number and the number of blocks as parameters and sets those blocks as free.
We calculate the index into which we need to go in the bitMapPointer array based on the startBlock value and start setting the bits as zero incrementally from that bit to until the requested number of blocks value becomes zero.

<u>void setTheBlocks(int startBlock, int numberOfBlocks);</u>

This function takes the starting block number and the number of blocks as parameters and sets those blocks as free.
We calculate the index into which we need to go in the bitMapPointer array based on the startBlock value and start setting the bits as one incrementally from that bit to until the requested number of blocks value becomes zero.

<u>int allocateFreeSpace(int size);</u>

This function takes the size in bytes as a parameter. We calculate the blocks that would be required based on the size we received.

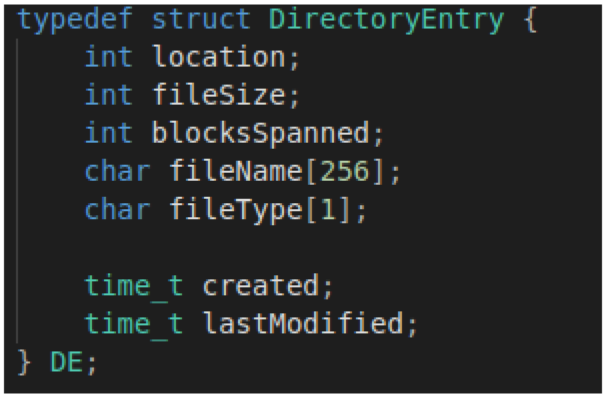
Now we start searching the bits from the beginning in the bitMapPointer array sequentially until we find the required number of consecutive free bits (0 bits).

If we encounter any bit with value 1, we reset the search, update our returnBlockNumber value with the current block number and start the search for consecutive free bits as per the requirement.

This way, if we ever find the required consecutive blocks, we will have the returnBlockNumber variable with the start block number value from which the required free blocks are available.

===

The Directory System:

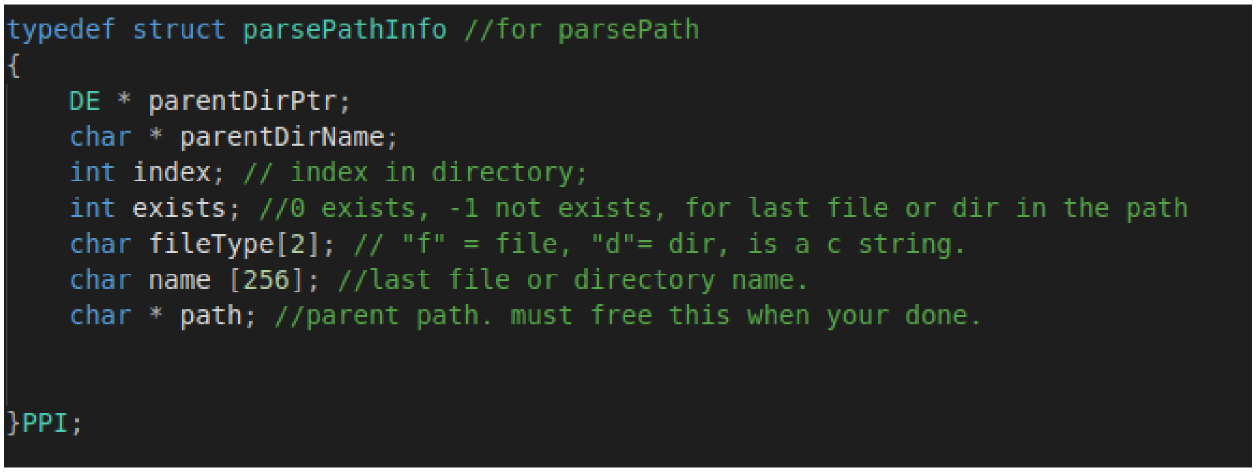


```
typedef struct DirectoryEntry {
    int location;
    int fileSize;
    int blocksSpanned;
    char fileName[256];
    char fileType[1];

    time_t created;
    time_t lastModified;
} DE;
```

Our Directory system will have a root node with “.” first directory Entry and “..” second directory entry followed by 8 other directory entries for files or Directories. We plan to expand the number of directory entries a directory can hold in the future. In the future we plan to identify directory entries that are not in use, currently they are just empty.

<u>Cd Main Functions for Our File System</u>:

ParsePath:



```
typedef struct parsePathInfo //for parsePath
{
    DE * parentDirPtr;
    char * parentDirName;
    int index; // index in directory;
    int exists; //0 exists, -1 not exists, for last file or dir in the path
    char fileType[2]; // "f" = file, "d"= dir, is a c string.
    char name [256]; //last file or directory name.
    char * path; //parent path. must free this when your done.


}PPI;
```

ParsePath is a function that needed to be implemented as quickly as possible because almost all the other functions rely on parsePath to work, so without parsepath it was difficult for our team to test the other functions. To implement a parsePath I divided tasks into dealing with an absolute path and a relative path. If it was an absolute path the path would start with a “/” and without a slash we are dealing with a relative path. The path was tokenized with “/” as a delimiter, so each individual token can be searched.

When dealing with an absolute path, root had to be loaded into memory and a pointer to root was stored in the PPI (parse path info) struct used by parse path to store information to be passed to other functions. After root is loaded, the directory is searched for the next token in the path, and if that exists and is a directory, the next token in the path is loaded until we get to the last token. The information about the last token in the path and the parent directory of the last token is stored in the PPI struct.

The PPI struct evolved over the course of the project to contain more fields as the project progressed. The members of the PPI struct are a pointer to the parent directory of the last name in the path, the index of the last file/dir in the parent directory, whether it exists or not, whether it's a file or directory, the name of the last token, and the absolute path to the parent directory. I realized that I could have just left the index member -1 if the last token was not found instead of having a member variable exist to tell users if the last token exists, but it would have been a lot of code to change and possibly introduce more bugs so we kept the redundancy.

Similarly, when a relative path was passed to parse path the PPI struct points to the current working directory in memory, so no need to load the first directory, and just like with the absolute path, load each consecutive token checking if it exists or is a directory up until the last token.

One tricky thing about our parse path function is that I chose to also build an absolute path string associated with the path passed, so users could compare paths to quickly determine if they needed to reload the current directory. Also getting the current working directory could quickly have a string to pass if the current directory also had an absolute path associated with it. This part of the code has been a continuous challenge. Thought it would be easier for everyone else to do their work if it had these multiple functions, but in retrospect would probably have saved a lot of time if we did not go that route.

fs_mkdir:

This function is used to make a directory in the location based on the path specified in the function parameter. I created a helper function that makes a directory when a pointer to a directory entry is passed. The directory passed to the helper function is given from the PPI struct returned from parse path. We search the parent directory to be for an empty directory entry. Once an empty directory entry is found, we ask our allocate frees space function to find a location with the number of free blocks that are requested. We have set a default value of fifty directory entries per directory. After getting a valid location from our free map that can accommodate our directory, a directory pointer is created and malloced to hold the default amount of directory entries. Then we can initialize our values for the “.” entry to the default values of a new directory, the starting location, size, blockspanned, and whether it's a directory or a file. Then the “..” directory values are set from the parent directory pointer passed to the make directory functions, that we get from the ‘.’ directory entry of the parent directory. Then since the directory is slightly smaller than the space in blocks allocated, we create a buffer of the

size equal to the number of blocks that our directory will span in our volume. Then the directory structure is memcpy to the buffer, then we do an lbaWrite to our volume. After the write the bytes representing the location and blocks spanned of our directory are marked used with the set as used function to flip the bits in our free space map for the appropriate location. After the directory is created we reload the current directory which we call dir to reflect the changes. We have a path string that's returned by PPI (parse path struct) that we match to the path of the current directory to see if it needs to be reloaded to reflect the changes of the new directory.
In hindsight to be easier on ourselves I would just reload the current directory with doing any checks.

fs_opendir:

This function is used to open the directory that we want to work on. It is used in various commands such as “ls” and also in various other directory functions such as readdir and rmdir.

Here is how I approached writing the fs_opendir:

Before we start opening the directory we create an array of type DE with length equivalent to the value of NUM_DE_IN_DIR.
fs_opendir function takes pathname as parameter. We pass the pathname to a helper routine fileOrDir to confirm if the given pathname is a directory.
We now call the parsePath on the pathname to get the pointer to the parent directory.

We create a fdDir pointer variable to hold the details of the directory that we are opening in this function and malloc the size equivalent to the fdDir structure.

We later create a buffer and LBAread into the buffer the required number of blocks, from the start location of the directory. We get the start location of the directory from the DE struct (PPI struct has a pointer of DE struct) as follows:
parentDir->parentDirPtr[parentDir->index].location

LBAread returns the number of blocks read, and we compare this value with the block count value we gave it to the LBAread to make sure everything is read.

Now we memcpy the contents of the buffer to the DE type array we have created in the beginning.

As a last step we set the values of the members of the struct fdDir through a variable we have created earlier and return the fdDir pointer variable.

fs_readdir:

fs_readdir takes in a pointer from fdDir struct and returns a pointer to the fs_diriteminfo structure. In this function, we we load the members of fs_diriteminfo with the appropriate information for a directories record length, file type, and the directory name. In order to get the appropriate information, we must first allocate memory to a pointer to fs_diriteminfo which we will populate. We must iterate through each of our directories and get each of the file names and file size and store them in the fs_diriteminfo structure. We use an integer index and set it equal to fdDir's directory entry position and use it to iterate through our directories. Once we iterate through all of our file systems directory and store our directories information inside of the diriteminfo structure, we return a pointer to fs_diriteminfo.

fs_closedir:

In fs_closedir, we free the memory allocated to our open directory pointer used in fs_opendir and reset it to null.

fs_getcwd:

This function prints the current working directory for the print working directory command. It takes the current working directory which is set by fs_setcwd and returns a pointer to the current working directory.

fs_setcwd:

This function sets our current working directory for change directory command. We first create a pointer to our parse path info and set it equal to our parse path function. This allows us to check whether or not a directory exists and if the path passed is a file or directory. If the type of the path passed is a directory, we call LBAread to read in the directory into a buffer from the location of the directory we want to set. We need to We then copy the contents of the buffer into the current directory to update the current directory. Then string copy the path into our current working directory pointer. If ".." is passed, we copy the parent directory into our current working directory.

fs_isFile:

This function is used in various shell functions to know if the used argument in the function call is a file or not. It is a simple function in which we call a helper function fileOrDir to tell us if the given filename is a file or not.

fs_isDir:

This function is used in various shell functions to know if the used argument in the function call is a directory or not. It is a simple function in which we call a helper function fileOrDir to tell us if the given pathname is a directory or not.

fs_rmdir:

This function takes in path as an argument. To check the validity of the path, parsePath is called and its return value is compared with null to check if the directory exists or not. We compare the value of fileType present in the particular directory entry to check if it is actually a directory and not any other file.

The function fs_opendir is called to check the contents of the directory to actually check if the directory is empty or not. So the approach begins with a pointer to the directory and incrementing it until with get an empty directory entry. We will keep incrementing the pointer and if a directory entry with a file is found, then we will return from the function with an error code.

If finally the directory is empty, we will free the blocks over which the directory is spanned and set the values in the directory entry to default.

fs_delete:

The approach to this function begins with calling parsePath to check the validity of the path argument passed in the command line. The return of parsePath is checked if it is null or not and returned from the function accordingly. If the return is a valid path, then the directory entry's values are set to default and the blocks occupied by the particular file are freed.

fs_stat:

The fs_stat function copies the contents of the particular directory entry, important properties like date created, date modified, file size and blocks spanned. Before the entries are copied, the parsePath function is called to check the validity of the path like if it is null or not. The value is null if the given path does not exist.

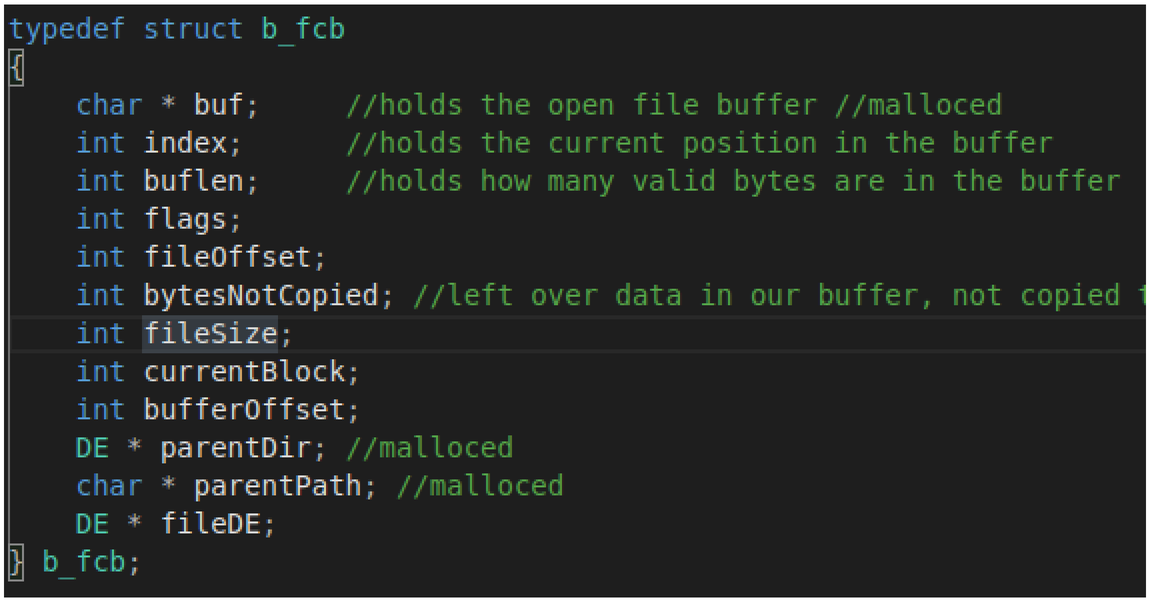
Move file command:

We first check if the number of arguments that we received is equal to 3 or not.
If yes, we check if the second argument is file and then the third argument is directory.
After all these checks evaluate to true, we create a buffer and LBAread the directory which is 3rd argument to the buffer and memcpy contents of this buffer to the DE pointer variable.
We now find the empty directory and save the address of that to a pointer variable.
Now we copy the directory entry details from the current directory to the destination directory to the return of the empty directory function pointer variable.
Now set the values in the current directory entry to default values.
We now LBAwrite the changes in both the directories.
At last we reload the directory to reflect the changes.

==

Buffered Input/Output functions:

b_open:



```
typedef struct b_fcb
{
    char * buf;     //holds the open file buffer //malloced
    int index;      //holds the current position in the buffer
    int buflen;     //holds how many valid bytes are in the buffer
    int flags;
    int fileOffset;
    int bytesNotCopied; //left over data in our buffer, not copied t
    int fileSize;
    int currentBlock;
    int bufferOffset;
    DE * parentDir; //malloced
    char * parentPath; //malloced
    DE * fileDE;
} b_fcb;
```

First we use parse path to get information on the file we are opening. The PPI struct parse path returns tells us if the file exists, its a directory or file, etc. We then ask for a free file control block. Then we populate the members of the FCB struct so the read, write, seek, and close functions can share important information about the file. If the file does exist we need to check if the truncate flag is passed, if so we set all of the values in the FCB struct regarding size and offset to 0, so the file is essentially erased. If the file does not exist we have to create a new file, and using the parent pointer from the PPI struct we can search for a free directory entry, and populate it with the new information for the created file, only if the O_creat flag is on. Otherwise we return NULL because we are not allowed to create a file.

b_seek:

This function takes file descriptor, offset and an int value called whence which describes from where should we move the offset to.

We have a switch statement based on whence value. This switch statement has values, SEEK_SET, SEEK_CUR, SEEK_END.

If the whence value is SEEK_SET, we update the offset to the value of offset from the parameter.

If the whence value is SEEK_CUR, we update the offset to the current offset value + offset value from the parameter.

If the whence value is SEEK_END, we update the offset to the filesize - offset value from the parameter.

b_write:

This function takes in a file descriptor, a buffer and a count variable which tells the function how many bytes it is supposed to write to the file pointed by the file descriptor.

The approach to writing this function included first checking if the source file's size is greater than the total size of the destination file (i.e., blocks spanned by the file times size of one block). If it was, then we would need to move the entire destination file to a new location with twice the number of blocks spanned initially and set the initial blocks as free.

If the source file's size was greater than or equal to how many blockswe allocated to a file times the bytes per block, we need to make the file size larger in order to write to the the destination file. In this case, we would double the blocks spanned. In order to preform this, we need to ask our free space to allocate double the blocks spanned and store the return as the new location we are going to write to. We must then copy the old contents of the file that we wrote to disk to the new location with the appropriate size of blocks spanned for the file and set the blocks in the new location as used and free the old blocks that were used originally. Finally, we reload the current directory to ensure nothing was lost.

Implementing write was difficult. Initially we tried to divide up the parts in which we write into two cases. The first case is if the count requested is greater than or equal to 512 size chunks and if count is less than or equal to 512 size chunks. We found a solution by creating a while loop that checks if there are still bytes remaining to be written. In this edge case, We must keep track of the files offset and the block offset. We use our helper routine blocksneeded which which we pass in the files offset - 1 which calculates our block offset which we used to write to disk. For our file system, we read and write 512 chunk sizes and in order to do so, we must increment our block offset when file offset is divisible by 512 and it is greater than 512. We must

LBAread the initial content into a temporary buffer that we have declared. If the file offset is less that 512 we can set the buffer location to the file offset else, we set the buffer location to the file offset mod 512. We must also calculate the bytes left to write by checking if the bytes remaining is greater than 512 minus the buffer location. We then copy the contents in the system buffer into our temporary buffer and increment the system buffer. We now can LBAwrite the contents in our temporary buffer one block at at time starting form the block offset plus the files location. We can now increment the files offset and the bytes written by the bytes to write and decrement the bytes remaining.

b_read:

For b_read, before we started to read, we have placed few checks such as to check if we reached EOF and changing the count value if the user asked to read bytes more than the remaining bytes in the file, then we have set the count value to the remaining bytes in the file. After that the read operations start and they are divided into 3 cases: 1) First case is if the count value (characters to read) is less than or equal to the bytes left in our buffer. In this case, we just copy the requested data from our buffer to the user buffer, without actually reading from the file a new block. 2) Second case is if we already have some data in buffer, and after subtracting that number of bytes the character to read is still less than the block size or if our buffer is empty and the user requested to read the data less than the block size bytes. 3) Third case is when the user requested more bytes than what we have in buffer which is also greater than the block size or when buffer is empty and user requested to read bytes greater than the block size. Based on each case, the way we fulfill the user's read request will be a bit different.

b_close:

For b_close, we free the allocated buffer to that file to perform the read operation.

How we worked together, how often we met, how we met, and how we divided up the tasks:

We have been working together as a team. No one has been shy to bring up issues they have faced and questions they needed answers to. Since the release of the file system project, we have always met after class to discuss and plan ideas for the project for about 10 to 15 minutes. For the past two weeks, we have done multiple zoom meetings to discuss the progress we have each made and potential ideas to improve our system. We have also met in the library common room to work on our project together utilizing the whiteboards to map out code. We divided up the work in pairs and worked together on the project as a whole.

Issues & Resolutions:

Issues faced while working on free space management:

I found it difficult to work with bits and manipulate them as per our need. For that reason, I took time to rewatch professor's lectures and other useful material on the internet to learn more on how to work with bits and manipulate them. After that, I started writing some useful helper routines to manipulate the bits.

Initially could not return the exact block number from which the free space allocation would start. To resolve this issue, I had to use a lot of print statements to understand how my code is behaving and what the values are at each iteration. Once I understood how the code was behaving, I had to make a couple of tweaks resetting the value of the requested number of blocks everytime we encountered a bit 1. After such small tweaks, I could return the right value to the user.

We faced a bunch of segmentation faults. To solve this issue, we used valgrind and added print statements to debug the memory issues.

We had calculation issues while trying to implement the root directory. To solve this issue we used print statements to print the size of the directory entry structure and carefully calculated the number of directory entries needed.

For some of us, it was our first time using github to work on a project with group members. We encountered merge conflicts. Thankfully, Pranjal is a github expert and was able to help us fix errors when trying to push and pull. We scheduled meetings to resolve all the conflicts using stash, drop, and merge git commands.

Volume Control Block:
Puting the volume control block together was not too hard, unfortunately we needed several other pieces of the project like the freeSpace bitMap, a way to allocate free space for the root directory and the root directory itself. We ended up hard coding most

of the VCB initially to get the program up and running, and as we finished parts that were necessary we added them on.

Root Directory:
Like the Volume control block we needed a free block allocation routine to tell us where there is free space to put the root directory. While the free space allocation routine was being developed we hard coded the values in the root directory, like starting location. When the memory allocation routine was ready we were getting the wrong location for the first free block of memory to write the root directory too. We thought it was the memory allocation routine because when our root was supposed to start at 6 it was instead starting at 48. We thought 8*6 is 48 so we must be having some issues with bytes being confused with bits. Turns out our free space map was marking a byte = 0x01 as being full, that only means the 8th bit or 8th block in that byte is not empty. Luckily it was a quick fix after figuring that out.

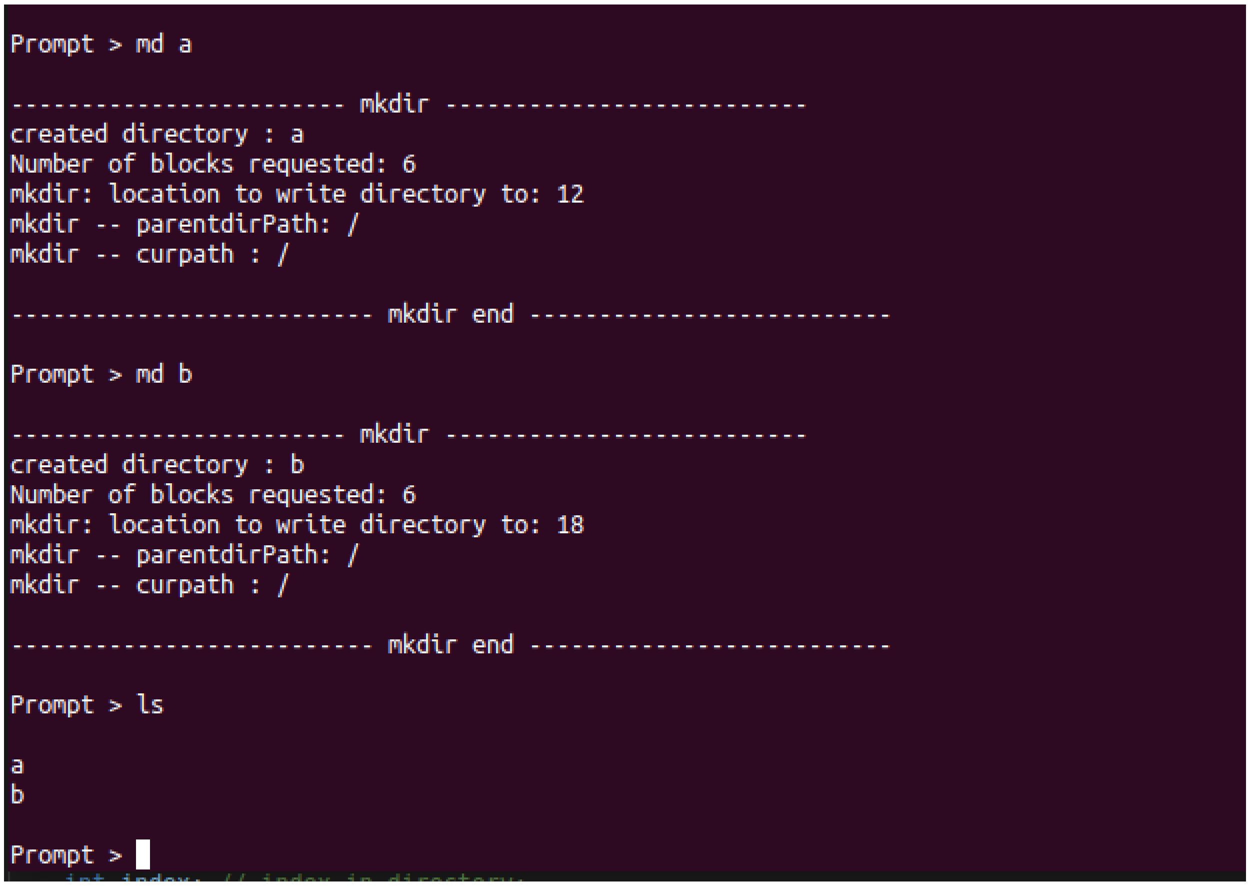
Buffered I/O functions:
We were having a hard time figuring out b_write method as everyone was concerned with corner cases. There were many issues that occurred during implementation as well as testing like nothing being read in cat due to file size not being updated, not handling the file offset properly, and much more. One of the most difficult cases we solved was when the count is greater than the block size, we had to keep track of the current block and index number of bytes in the block. Once we had gotten this working, we tested out if the file size was greater than the number of blocks spanned by the destination file. If it was, then we implemented a way to move the file to a location such that there will be free blocks twice the initial number to successfully store the contents in one flow.

mv command issues:

The main difficulty was to LBAwrite the changed directory entry to the right location. After using a lot of print statements, it worked at the end.

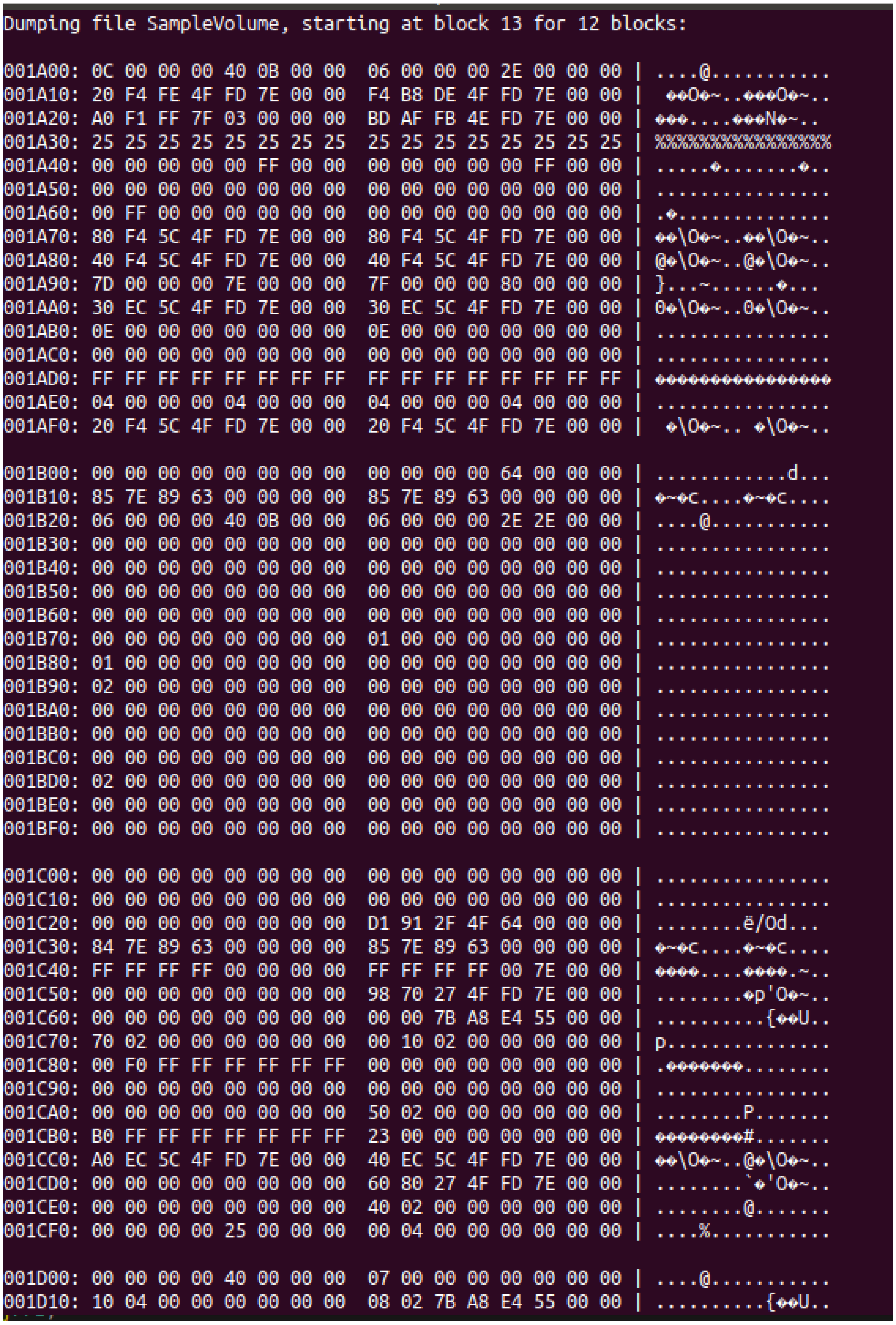
Screenshots of Our File System:

cmd_md and cmd_ls :

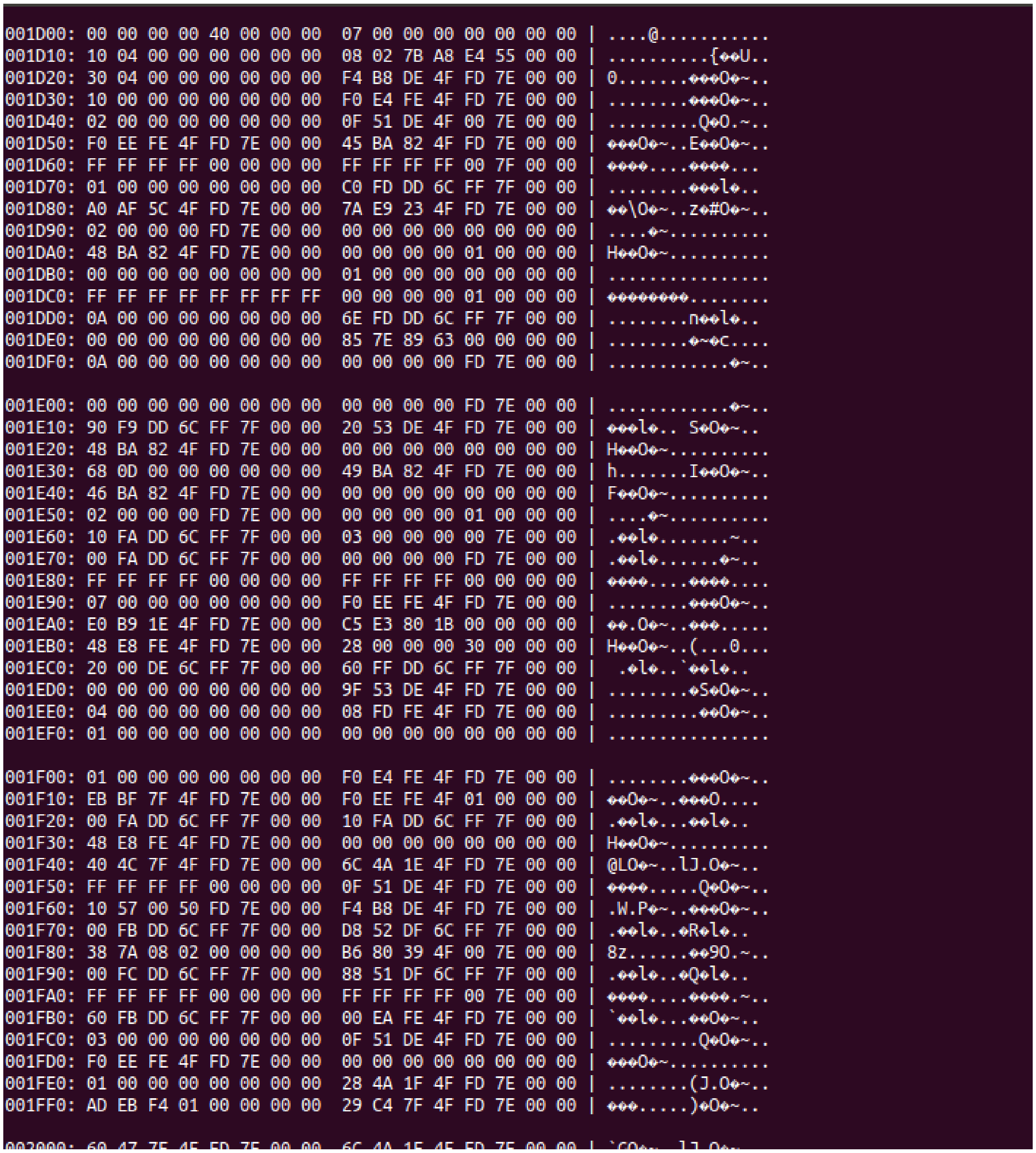


```
Prompt > md a

------------------------ mkdir -------------------------
created directory : a
Number of blocks requested: 6
mkdir: location to write directory to: 12
mkdir -- parentdirPath: /
mkdir -- curpath : /

-------------------------- mkdir end --------------------------

Prompt > md b

------------------------ mkdir -------------------------
created directory : b
Number of blocks requested: 6
mkdir: location to write directory to: 18
mkdir -- parentdirPath: /
mkdir -- curpath : /

-------------------------- mkdir end --------------------------

Prompt > ls

a
b

Prompt >
```

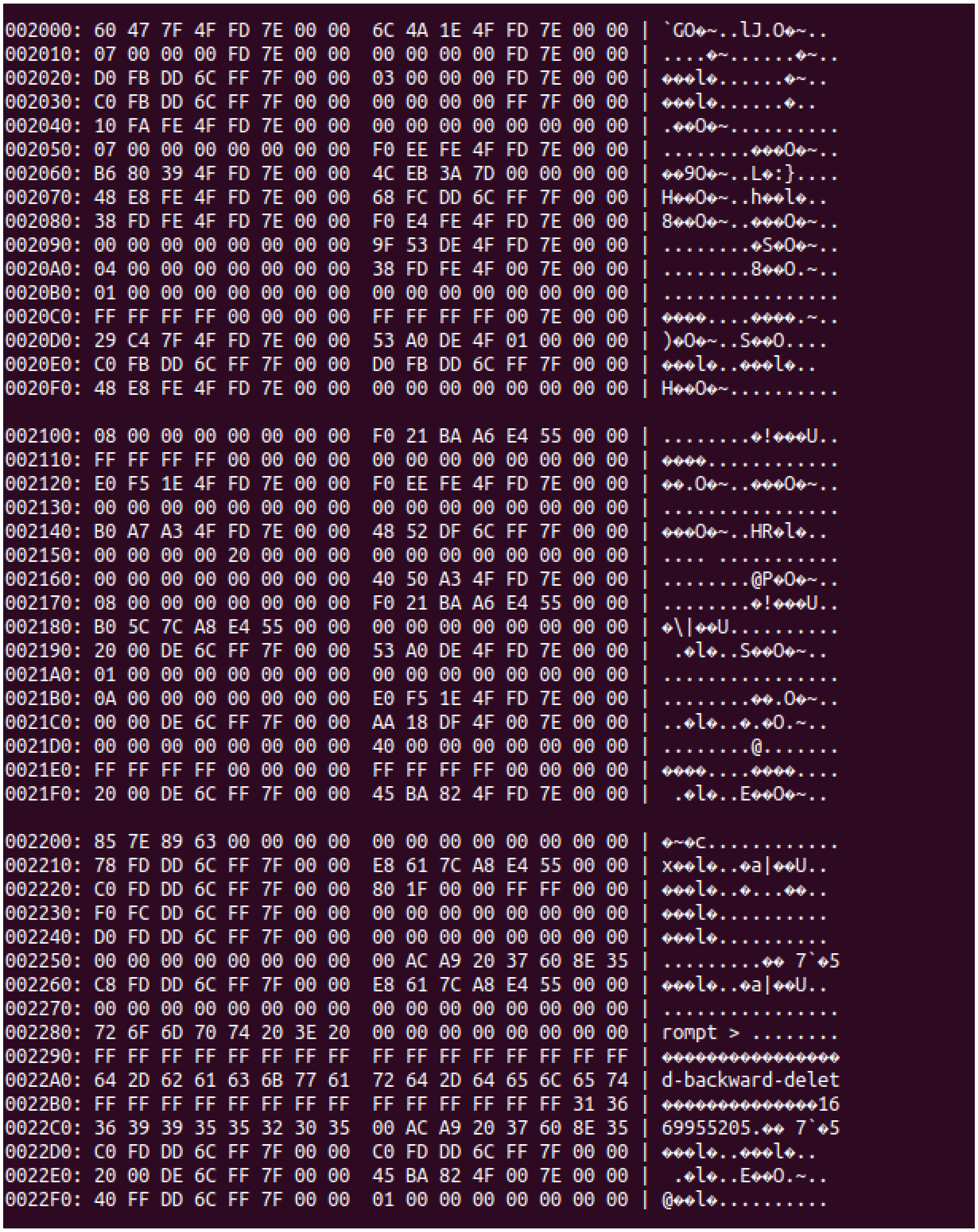
Hexdump of md a:



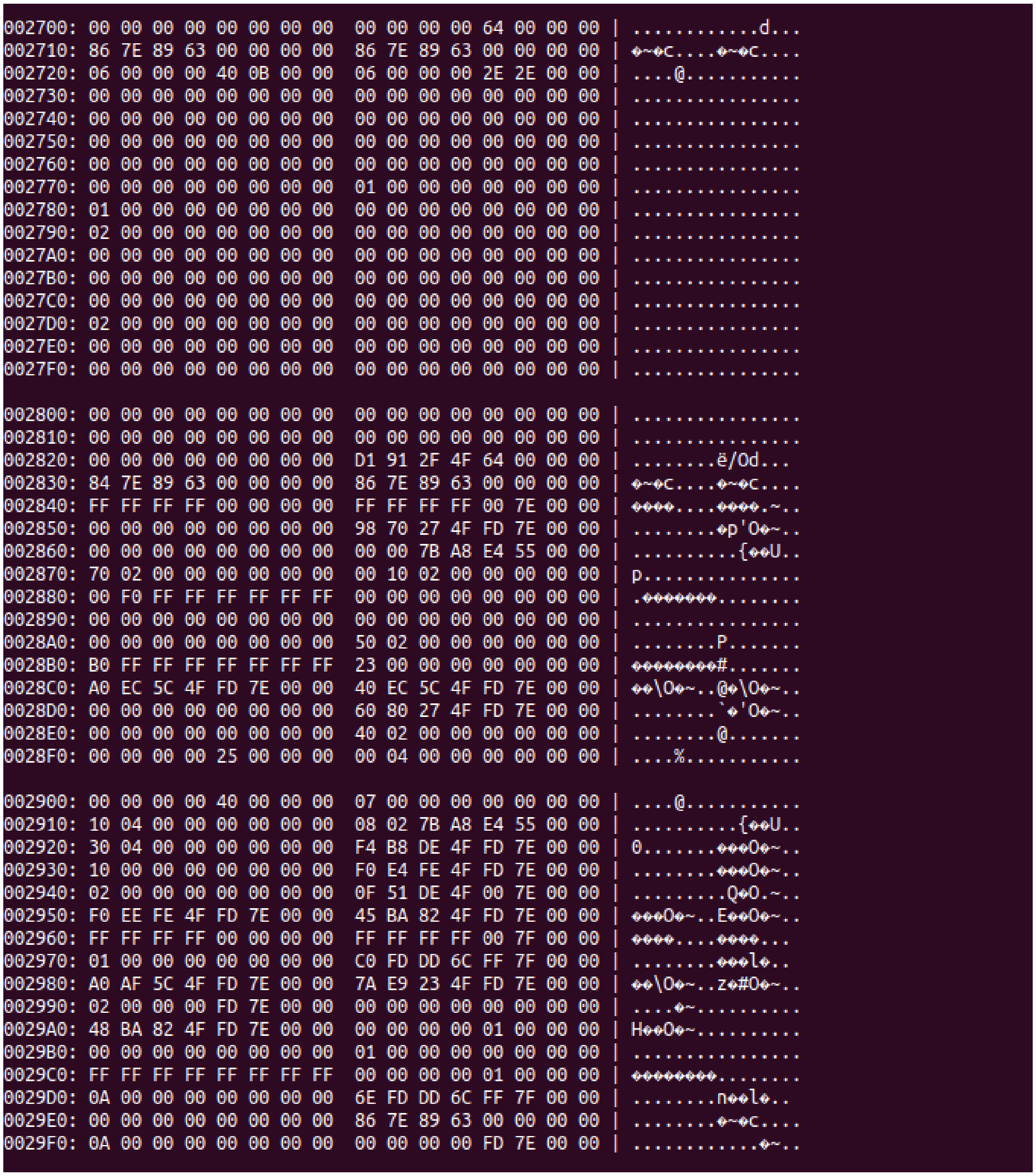
```
Dumping file SampleVolume, starting at block 13 for 12 blocks:

001A00: 0C 00 00 00 40 0B 00 00  06 00 00 00 2E 00 00 00 | ....@...........
001A10: 20 F4 FE 4F FD 7E 00 00  F4 B8 DE 4F FD 7E 00 00 |  ��O�~..���O�~..
001A20: A0 F1 FF 7F 03 00 00 00  BD AF FB 4E FD 7E 00 00 | ���....���N�~..
001A30: 25 25 25 25 25 25 25 25  25 25 25 25 25 25 25 25 | %%%%%%%%%%%%%%%%
001A40: 00 00 00 00 00 FF 00 00  00 00 00 00 00 FF 00 00 | .....�.......�..
001A50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001A60: 00 FF 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | .�..............
001A70: 80 F4 5C 4F FD 7E 00 00  80 F4 5C 4F FD 7E 00 00 | ��\O�~..��\O�~..
001A80: 40 F4 5C 4F FD 7E 00 00  40 F4 5C 4F FD 7E 00 00 | @�\O�~..@�\O�~..
001A90: 7D 00 00 00 7E 00 00 00  7F 00 00 00 80 00 00 00 | }...~.......�...
001AA0: 30 EC 5C 4F FD 7E 00 00  30 EC 5C 4F FD 7E 00 00 | 0�\O�~..0�\O�~..
001AB0: 0E 00 00 00 00 00 00 00  0E 00 00 00 00 00 00 00 | ................
001AC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001AD0: FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF | ����������������
001AE0: 04 00 00 00 04 00 00 00  04 00 00 00 04 00 00 00 | ................
001AF0: 20 F4 5C 4F FD 7E 00 00  20 F4 5C 4F FD 7E 00 00 |  �\O�~.. �\O�~..

001B00: 00 00 00 00 00 00 00 00  00 00 00 00 64 00 00 00 | ............d...
001B10: 85 7E 89 63 00 00 00 00  85 7E 89 63 00 00 00 00 | �~�c....�~�c....
001B20: 06 00 00 00 40 0B 00 00  06 00 00 00 2E 2E 00 00 | ....@...........
001B30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001B40: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001B50: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001B60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001B70: 00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00 | ................
001B80: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001B90: 02 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BB0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BC0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BD0: 02 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BE0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001BF0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

001C00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001C10: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001C20: 00 00 00 00 00 00 00 00  D1 91 2F 4F 64 00 00 00 | ........ë/Od...
001C30: 84 7E 89 63 00 00 00 00  85 7E 89 63 00 00 00 00 | �~�c....�~�c....
001C40: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
001C50: 00 00 00 00 00 00 00 00  98 70 27 4F FD 7E 00 00 | ........�p'O�~..
001C60: 00 00 00 00 00 00 00 00  00 00 7B A8 E4 55 00 00 | ..........{��U..
001C70: 70 02 00 00 00 00 00 00  00 10 02 00 00 00 00 00 | p...............
001C80: 00 F0 FF FF FF FF FF FF  00 00 00 00 00 00 00 00 | .�������........
001C90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
001CA0: 00 00 00 00 00 00 00 00  50 02 00 00 00 00 00 00 | ........P.......
001CB0: B0 FF FF FF FF FF FF FF  23 00 00 00 00 00 00 00 | ��������#.......
001CC0: A0 EC 5C 4F FD 7E 00 00  40 EC 5C 4F FD 7E 00 00 | ��\O�~..@�\O�~..
001CD0: 00 00 00 00 00 00 00 00  60 80 27 4F FD 7E 00 00 | ........`�'O�~..
001CE0: 00 00 00 00 00 00 00 00  40 02 00 00 00 00 00 00 | ........@.......
001CF0: 00 00 00 00 25 00 00 00  00 04 00 00 00 00 00 00 | ....%...........

001D00: 00 00 00 00 40 00 00 00  07 00 00 00 00 00 00 00 | ....@...........
001D10: 10 04 00 00 00 00 00 00  08 02 7B A8 E4 55 00 00 | ..........{��U..
```



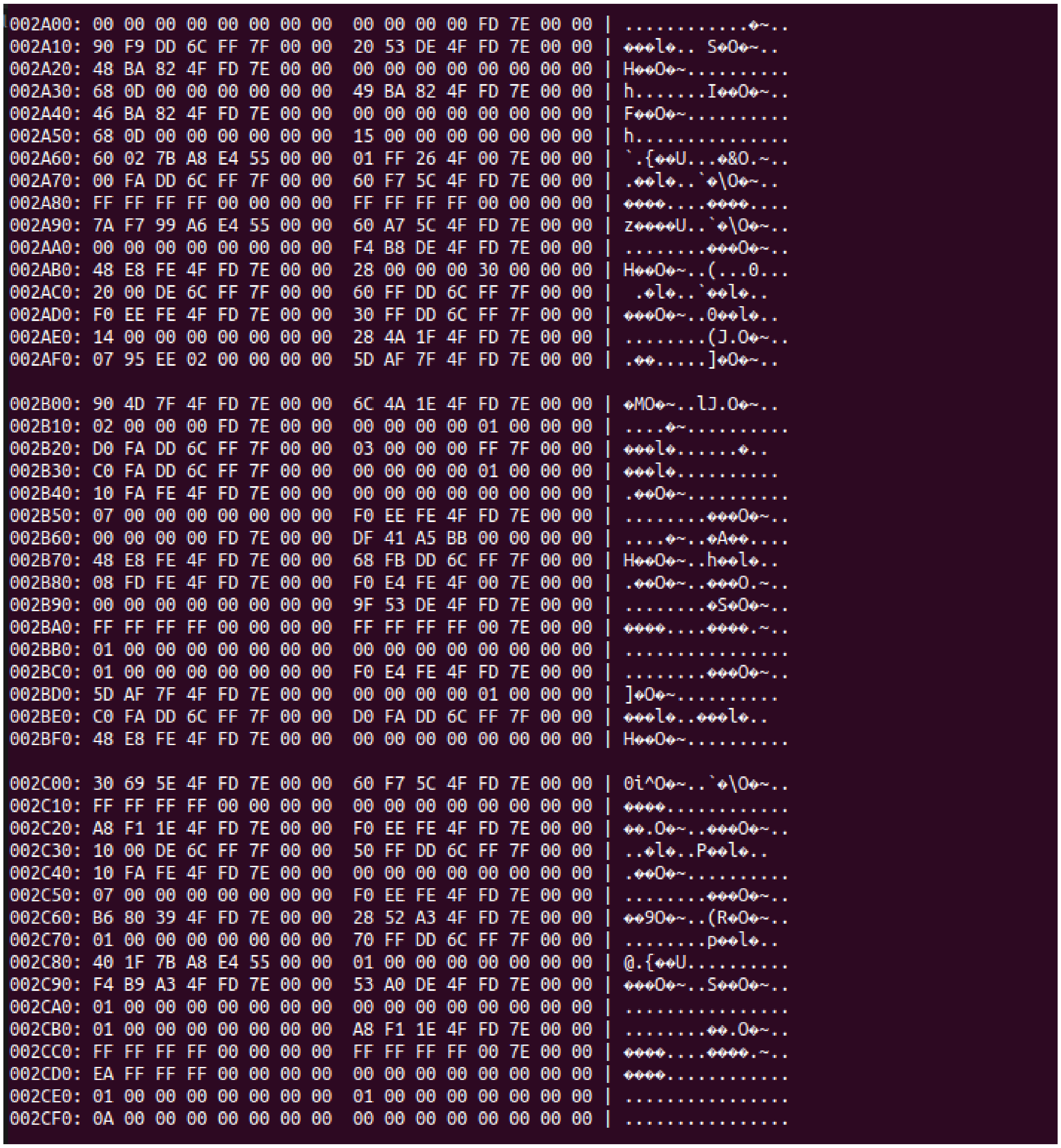
```
001D00: 00 00 00 00 40 00 00 00  07 00 00 00 00 00 00 00 | ....@...........
001D10: 10 04 00 00 00 00 00 00  08 02 7B A8 E4 55 00 00 | ..........{◆◆U..
001D20: 30 04 00 00 00 00 00 00  F4 B8 DE 4F FD 7E 00 00 | 0.......◆◆◆O◆~..
001D30: 10 00 00 00 00 00 00 00  F0 E4 FE 4F FD 7E 00 00 | ........◆◆◆O◆~..
001D40: 02 00 00 00 00 00 00 00  0F 51 DE 4F 00 7E 00 00 | .........Q◆O.~..
001D50: F0 EE FE 4F FD 7E 00 00  45 BA 82 4F FD 7E 00 00 | ◆◆◆O◆~..E◆◆O◆~..
001D60: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7F 00 00 | ◆◆◆◆....◆◆◆◆...
001D70: 01 00 00 00 00 00 00 00  C0 FD DD 6C FF 7F 00 00 | ........◆◆◆l◆..
001D80: A0 AF 5C 4F FD 7E 00 00  7A E9 23 4F FD 7E 00 00 | ◆◆\O◆~..z◆#O◆~..
001D90: 02 00 00 00 FD 7E 00 00  00 00 00 00 00 00 00 00 | ....◆~..........
001DA0: 48 BA 82 4F FD 7E 00 00  00 00 00 00 01 00 00 00 | H◆◆O◆~..........
001DB0: 00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00 | ................
001DC0: FF FF FF FF FF FF FF FF  00 00 00 00 01 00 00 00 | ◆◆◆◆◆◆◆◆........
001DD0: 0A 00 00 00 00 00 00 00  6E FD DD 6C FF 7F 00 00 | ........n◆◆l◆..
001DE0: 00 00 00 00 00 00 00 00  85 7E 89 63 00 00 00 00 | ........◆~◆c....
001DF0: 0A 00 00 00 00 00 00 00  00 00 00 00 FD 7E 00 00 | ............◆~..

001E00: 00 00 00 00 00 00 00 00  00 00 00 00 FD 7E 00 00 | ............◆~..
001E10: 90 F9 DD 6C FF 7F 00 00  20 53 DE 4F FD 7E 00 00 | ◆◆◆l◆.. S◆O◆~..
001E20: 48 BA 82 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | H◆◆O◆~..........
001E30: 68 0D 00 00 00 00 00 00  49 BA 82 4F FD 7E 00 00 | h.......I◆◆O◆~..
001E40: 46 BA 82 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | F◆◆O◆~..........
001E50: 02 00 00 00 FD 7E 00 00  00 00 00 00 01 00 00 00 | ....◆~..........
001E60: 10 FA DD 6C FF 7F 00 00  03 00 00 00 00 7E 00 00 | .◆◆l◆.......~..
001E70: 00 FA DD 6C FF 7F 00 00  00 00 00 00 FD 7E 00 00 | .◆◆l◆......◆~..
001E80: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ◆◆◆◆....◆◆◆◆....
001E90: 07 00 00 00 00 00 00 00  F0 EE FE 4F FD 7E 00 00 | ........◆◆◆O◆~..
001EA0: E0 B9 1E 4F FD 7E 00 00  C5 E3 80 1B 00 00 00 00 | ◆◆.O◆~..◆◆◆.....
001EB0: 48 E8 FE 4F FD 7E 00 00  28 00 00 00 30 00 00 00 | H◆◆O◆~..(...0...
001EC0: 20 00 DE 6C FF 7F 00 00  60 FF DD 6C FF 7F 00 00 |  .◆l◆..`◆◆l◆..
001ED0: 00 00 00 00 00 00 00 00  9F 53 DE 4F FD 7E 00 00 | ........◆S◆O◆~..
001EE0: 04 00 00 00 00 00 00 00  08 FD FE 4F FD 7E 00 00 | .........◆◆O◆~..
001EF0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

001F00: 01 00 00 00 00 00 00 00  F0 E4 FE 4F FD 7E 00 00 | ........◆◆◆O◆~..
001F10: EB BF 7F 4F FD 7E 00 00  F0 EE FE 4F 01 00 00 00 | ◆◆O◆~..◆◆◆O....
001F20: 00 FA DD 6C FF 7F 00 00  10 FA DD 6C FF 7F 00 00 | .◆◆l◆...◆◆l◆..
001F30: 48 E8 FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | H◆◆O◆~..........
001F40: 40 4C 7F 4F FD 7E 00 00  6C 4A 1E 4F FD 7E 00 00 | @LO◆~..lJ.O◆~..
001F50: FF FF FF FF 00 00 00 00  0F 51 DE 4F FD 7E 00 00 | ◆◆◆◆.....Q◆O◆~..
001F60: 10 57 00 50 FD 7E 00 00  F4 B8 DE 4F FD 7E 00 00 | .W.P◆~..◆◆◆O◆~..
001F70: 00 FB DD 6C FF 7F 00 00  D8 52 DF 6C FF 7F 00 00 | .◆◆l◆..◆R◆l◆..
001F80: 38 7A 08 02 00 00 00 00  B6 80 39 4F 00 7E 00 00 | 8z......◆◆9O.~..
001F90: 00 FC DD 6C FF 7F 00 00  88 51 DF 6C FF 7F 00 00 | .◆◆l◆..◆Q◆l◆..
001FA0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ◆◆◆◆....◆◆◆◆.~..
001FB0: 60 FB DD 6C FF 7F 00 00  00 EA FE 4F FD 7E 00 00 | `◆◆l◆...◆◆O◆~..
001FC0: 03 00 00 00 00 00 00 00  0F 51 DE 4F FD 7E 00 00 | .........Q◆O◆~..
001FD0: F0 EE FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | ◆◆◆O◆~..........
001FE0: 01 00 00 00 00 00 00 00  28 4A 1F 4F FD 7E 00 00 | ........(J.O◆~..
001FF0: AD EB F4 01 00 00 00 00  29 C4 7F 4F FD 7E 00 00 | ◆◆◆.....)◆O◆~..

002000: 60 47 7F 4F FD 7E 00 00  6C 4A 1F 4F FD 7E 00 00 | `GO◆~..lJ.O◆~
```



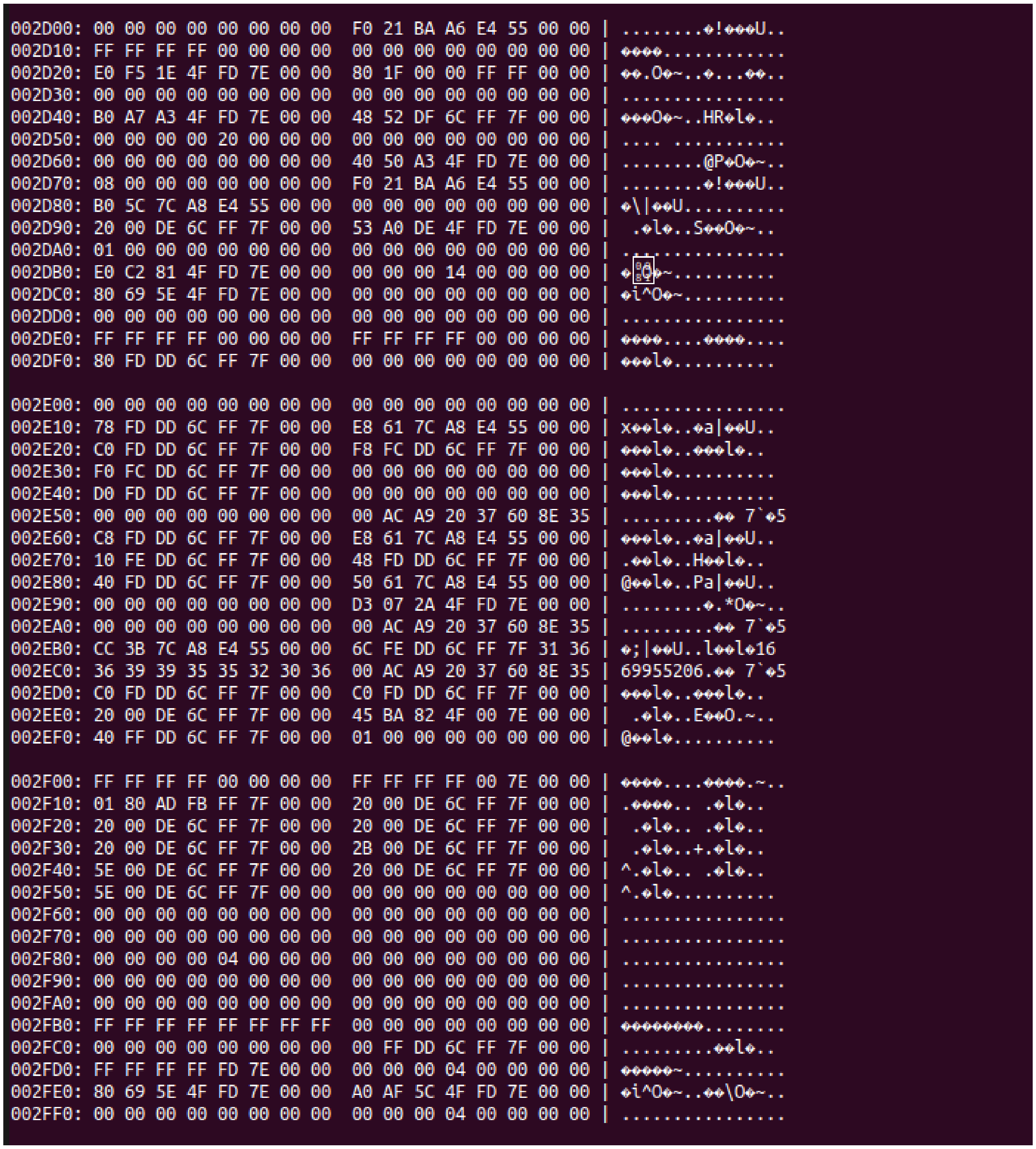
```
002000: 60 47 7F 4F FD 7E 00 00  6C 4A 1E 4F FD 7E 00 00 | `GO�~..lJ.O�~..
002010: 07 00 00 00 FD 7E 00 00  00 00 00 00 FD 7E 00 00 | ....�~......�~..
002020: D0 FB DD 6C FF 7F 00 00  03 00 00 00 FD 7E 00 00 | ���l�......�~..
002030: C0 FB DD 6C FF 7F 00 00  00 00 00 00 FF 7F 00 00 | ���l�......�..
002040: 10 FA FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | .��O�~..........
002050: 07 00 00 00 00 00 00 00  F0 EE FE 4F FD 7E 00 00 | ........���O�~..
002060: B6 80 39 4F FD 7E 00 00  4C EB 3A 7D 00 00 00 00 | ��9O�~..L�:}....
002070: 48 E8 FE 4F FD 7E 00 00  68 FC DD 6C FF 7F 00 00 | H��O�~..h��l�..
002080: 38 FD FE 4F FD 7E 00 00  F0 E4 FE 4F FD 7E 00 00 | 8��O�~..���O�~..
002090: 00 00 00 00 00 00 00 00  9F 53 DE 4F FD 7E 00 00 | ........�S�O�~..
0020A0: 04 00 00 00 00 00 00 00  38 FD FE 4F 00 7E 00 00 | ........8��O.~..
0020B0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0020C0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
0020D0: 29 C4 7F 4F FD 7E 00 00  53 A0 DE 4F 01 00 00 00 | )�O�~..S��O....
0020E0: C0 FB DD 6C FF 7F 00 00  D0 FB DD 6C FF 7F 00 00 | ���l�..���l�..
0020F0: 48 E8 FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | H��O�~..........

002100: 08 00 00 00 00 00 00 00  F0 21 BA A6 E4 55 00 00 | ........�!���U..
002110: FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ����............
002120: E0 F5 1E 4F FD 7E 00 00  F0 EE FE 4F FD 7E 00 00 | ��.O�~..���O�~..
002130: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002140: B0 A7 A3 4F FD 7E 00 00  48 52 DF 6C FF 7F 00 00 | ���O�~..HR�l�..
002150: 00 00 00 00 20 00 00 00  00 00 00 00 00 00 00 00 | .... ...........
002160: 00 00 00 00 00 00 00 00  40 50 A3 4F FD 7E 00 00 | ........@P�O�~..
002170: 08 00 00 00 00 00 00 00  F0 21 BA A6 E4 55 00 00 | ........�!���U..
002180: B0 5C 7C A8 E4 55 00 00  00 00 00 00 00 00 00 00 | �\|��U..........
002190: 20 00 DE 6C FF 7F 00 00  53 A0 DE 4F FD 7E 00 00 |  .�l�..S��O�~..
0021A0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0021B0: 0A 00 00 00 00 00 00 00  E0 F5 1E 4F FD 7E 00 00 | ........��.O�~..
0021C0: 00 00 DE 6C FF 7F 00 00  AA 18 DF 4F 00 7E 00 00 | ..�l�..�.�O.~..
0021D0: 00 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00 | ........@.......
0021E0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ����....����....
0021F0: 20 00 DE 6C FF 7F 00 00  45 BA 82 4F FD 7E 00 00 |  .�l�..E��O�~..

002200: 85 7E 89 63 00 00 00 00  00 00 00 00 00 00 00 00 | �~�c............
002210: 78 FD DD 6C FF 7F 00 00  E8 61 7C A8 E4 55 00 00 | x��l�..�a|��U..
002220: C0 FD DD 6C FF 7F 00 00  80 1F 00 00 FF FF 00 00 | ���l�..�...��..
002230: F0 FC DD 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ���l�..........
002240: D0 FD DD 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ���l�..........
002250: 00 00 00 00 00 00 00 00  00 AC A9 20 37 60 8E 35 | .........�� 7`�5
002260: C8 FD DD 6C FF 7F 00 00  E8 61 7C A8 E4 55 00 00 | ���l�..�a|��U..
002270: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002280: 72 6F 6D 70 74 20 3E 20  00 00 00 00 00 00 00 00 | rompt > ........
002290: FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF | ����������������
0022A0: 64 2D 62 61 63 6B 77 61  72 64 2D 64 65 6C 65 74 | d-backward-delet
0022B0: FF FF FF FF FF FF FF FF  FF FF FF FF FF FF 31 36 | ��������������16
0022C0: 36 39 39 35 35 32 30 35  00 AC A9 20 37 60 8E 35 | 69955205.�� 7`�5
0022D0: C0 FD DD 6C FF 7F 00 00  C0 FD DD 6C FF 7F 00 00 | ���l�..���l�..
0022E0: 20 00 DE 6C FF 7F 00 00  45 BA 82 4F 00 7E 00 00 |  .�l�..E��O.~..
0022F0: 40 FF DD 6C FF 7F 00 00  01 00 00 00 00 00 00 00 | @��l�..........
```



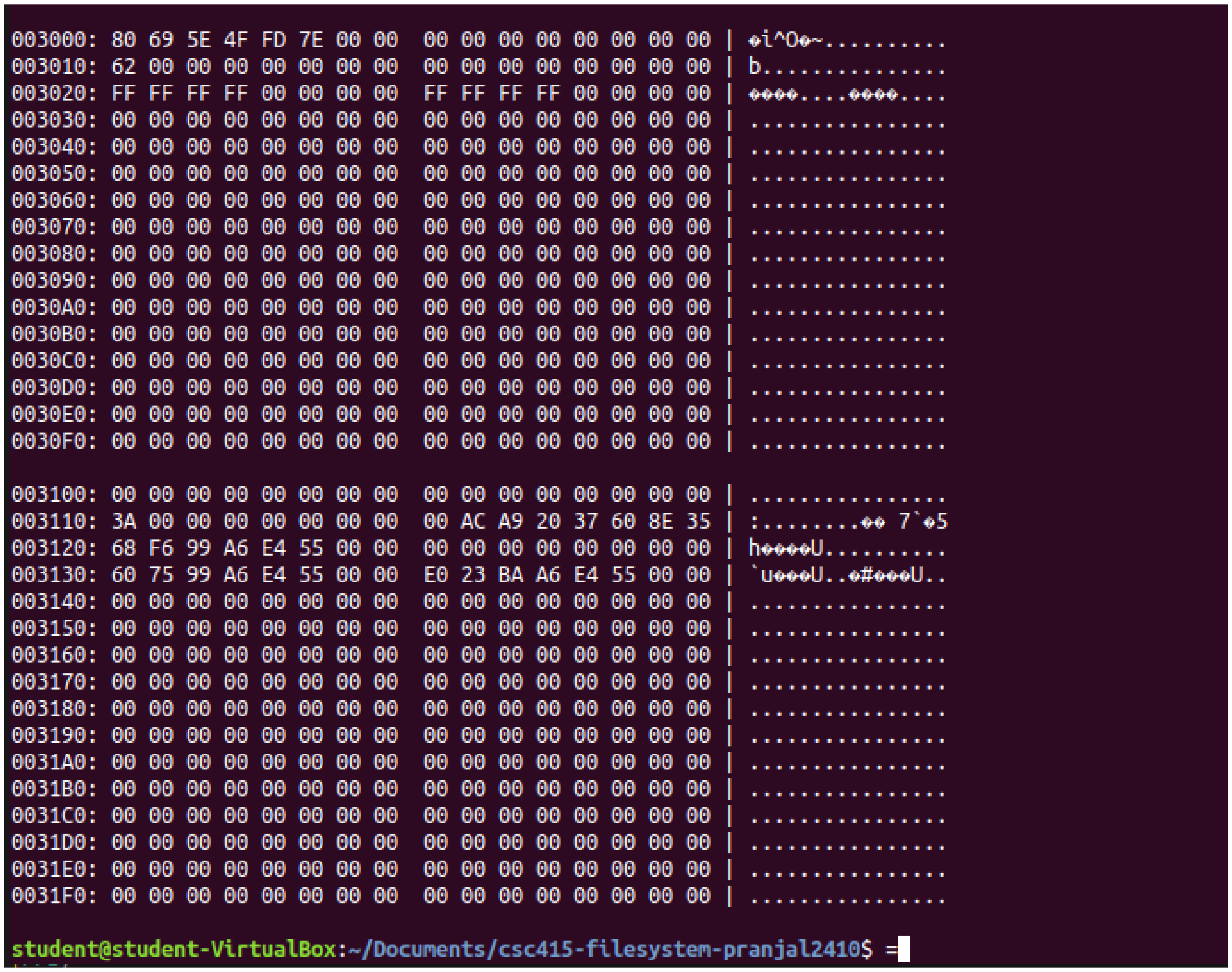
```
002700: 00 00 00 00 00 00 00 00  00 00 00 00 64 00 00 00 | ............d...
002710: 86 7E 89 63 00 00 00 00  86 7E 89 63 00 00 00 00 | �~�c....�~�c....
002720: 06 00 00 00 40 0B 00 00  06 00 00 00 2E 2E 00 00 | ....@...........
002730: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002740: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002750: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002760: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002770: 00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00 | ................
002780: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002790: 02 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027D0: 02 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0027F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

002800: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002810: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002820: 00 00 00 00 00 00 00 00  D1 91 2F 4F 64 00 00 00 | ........ë/Od...
002830: 84 7E 89 63 00 00 00 00  86 7E 89 63 00 00 00 00 | �~�c....�~�c....
002840: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
002850: 00 00 00 00 00 00 00 00  98 70 27 4F FD 7E 00 00 | ........�p'O�~..
002860: 00 00 00 00 00 00 00 00  00 00 7B A8 E4 55 00 00 | ..........{��U..
002870: 70 02 00 00 00 00 00 00  00 10 02 00 00 00 00 00 | p...............
002880: 00 F0 FF FF FF FF FF FF  00 00 00 00 00 00 00 00 | .�������........
002890: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0028A0: 00 00 00 00 00 00 00 00  50 02 00 00 00 00 00 00 | ........P.......
0028B0: B0 FF FF FF FF FF FF FF  23 00 00 00 00 00 00 00 | ��������#.......
0028C0: A0 EC 5C 4F FD 7E 00 00  40 EC 5C 4F FD 7E 00 00 | ��\O�~..@�\O�~..
0028D0: 00 00 00 00 00 00 00 00  60 80 27 4F FD 7E 00 00 | ........`�'O�~..
0028E0: 00 00 00 00 00 00 00 00  40 02 00 00 00 00 00 00 | ........@.......
0028F0: 00 00 00 00 25 00 00 00  00 04 00 00 00 00 00 00 | ....%...........

002900: 00 00 00 00 40 00 00 00  07 00 00 00 00 00 00 00 | ....@...........
002910: 10 04 00 00 00 00 00 00  08 02 7B A8 E4 55 00 00 | ..........{��U..
002920: 30 04 00 00 00 00 00 00  F4 B8 DE 4F FD 7E 00 00 | 0.......���O�~..
002930: 10 00 00 00 00 00 00 00  F0 E4 FE 4F FD 7E 00 00 | ........���O�~..
002940: 02 00 00 00 00 00 00 00  0F 51 DE 4F 00 7E 00 00 | .........Q�O.~..
002950: F0 EE FE 4F FD 7E 00 00  45 BA 82 4F FD 7E 00 00 | ���O�~..E��O�~..
002960: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7F 00 00 | ����....����...
002970: 01 00 00 00 00 00 00 00  C0 FD DD 6C FF 7F 00 00 | ........���l�..
002980: A0 AF 5C 4F FD 7E 00 00  7A E9 23 4F FD 7E 00 00 | ��\O�~..z�#O�~..
002990: 02 00 00 00 FD 7E 00 00  00 00 00 00 00 00 00 00 | ....�~..........
0029A0: 48 BA 82 4F FD 7E 00 00  00 00 00 00 01 00 00 00 | H��O�~..........
0029B0: 00 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00 | ................
0029C0: FF FF FF FF FF FF FF FF  00 00 00 00 01 00 00 00 | ��������........
0029D0: 0A 00 00 00 00 00 00 00  6E FD DD 6C FF 7F 00 00 | ........n��l�..
0029E0: 00 00 00 00 00 00 00 00  86 7E 89 63 00 00 00 00 | ........�~�c....
0029F0: 0A 00 00 00 00 00 00 00  00 00 00 00 FD 7E 00 00 | ............�~..
```



```
002A00: 00 00 00 00 00 00 00 00  00 00 00 00 FD 7E 00 00 | ............�~..
002A10: 90 F9 DD 6C FF 7F 00 00  20 53 DE 4F FD 7E 00 00 | ���l�.. S�O�~..
002A20: 48 BA 82 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | H��O�~..........
002A30: 68 0D 00 00 00 00 00 00  49 BA 82 4F FD 7E 00 00 | h.......I��O�~..
002A40: 46 BA 82 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | F��O�~..........
002A50: 68 0D 00 00 00 00 00 00  15 00 00 00 00 00 00 00 | h...............
002A60: 60 02 7B A8 E4 55 00 00  01 FF 26 4F 00 7E 00 00 | `.{��U...�&O.~..
002A70: 00 FA DD 6C FF 7F 00 00  60 F7 5C 4F FD 7E 00 00 | .��l�..`�\O�~..
002A80: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ����....����....
002A90: 7A F7 99 A6 E4 55 00 00  60 A7 5C 4F FD 7E 00 00 | z����U..`�\O�~..
002AA0: 00 00 00 00 00 00 00 00  F4 B8 DE 4F FD 7E 00 00 | ........���O�~..
002AB0: 48 E8 FE 4F FD 7E 00 00  28 00 00 00 30 00 00 00 | H��O�~..(...0...
002AC0: 20 00 DE 6C FF 7F 00 00  60 FF DD 6C FF 7F 00 00 |  .�l�..`��l�..
002AD0: F0 EE FE 4F FD 7E 00 00  30 FF DD 6C FF 7F 00 00 | ���O�~..0��l�..
002AE0: 14 00 00 00 00 00 00 00  28 4A 1F 4F FD 7E 00 00 | ........(J.O�~..
002AF0: 07 95 EE 02 00 00 00 00  5D AF 7F 4F FD 7E 00 00 | .��.....]�O�~..

002B00: 90 4D 7F 4F FD 7E 00 00  6C 4A 1E 4F FD 7E 00 00 | �MO�~..lJ.O�~..
002B10: 02 00 00 00 FD 7E 00 00  00 00 00 00 01 00 00 00 | ....�~..........
002B20: D0 FA DD 6C FF 7F 00 00  03 00 00 00 FF 7F 00 00 | ���l�......�..
002B30: C0 FA DD 6C FF 7F 00 00  00 00 00 00 01 00 00 00 | ���l�..........
002B40: 10 FA FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | .��O�~..........
002B50: 07 00 00 00 00 00 00 00  F0 EE FE 4F FD 7E 00 00 | ........���O�~..
002B60: 00 00 00 00 FD 7E 00 00  DF 41 A5 BB 00 00 00 00 | ....�~..�A��....
002B70: 48 E8 FE 4F FD 7E 00 00  68 FB DD 6C FF 7F 00 00 | H��O�~..h��l�..
002B80: 08 FD FE 4F FD 7E 00 00  F0 E4 FE 4F 00 7E 00 00 | .��O�~..���O.~..
002B90: 00 00 00 00 00 00 00 00  9F 53 DE 4F FD 7E 00 00 | ........�S�O�~..
002BA0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
002BB0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002BC0: 01 00 00 00 00 00 00 00  F0 E4 FE 4F FD 7E 00 00 | ........���O�~..
002BD0: 5D AF 7F 4F FD 7E 00 00  00 00 00 00 01 00 00 00 | ]�O�~..........
002BE0: C0 FA DD 6C FF 7F 00 00  D0 FA DD 6C FF 7F 00 00 | ���l�..���l�..
002BF0: 48 E8 FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | H��O�~..........

002C00: 30 69 5E 4F FD 7E 00 00  60 F7 5C 4F FD 7E 00 00 | 0i^O�~..`�\O�~..
002C10: FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ����............
002C20: A8 F1 1E 4F FD 7E 00 00  F0 EE FE 4F FD 7E 00 00 | ��.O�~..���O�~..
002C30: 10 00 DE 6C FF 7F 00 00  50 FF DD 6C FF 7F 00 00 | ..�l�..P��l�..
002C40: 10 FA FE 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | .��O�~..........
002C50: 07 00 00 00 00 00 00 00  F0 EE FE 4F FD 7E 00 00 | ........���O�~..
002C60: B6 80 39 4F FD 7E 00 00  28 52 A3 4F FD 7E 00 00 | ��9O�~..(R�O�~..
002C70: 01 00 00 00 00 00 00 00  70 FF DD 6C FF 7F 00 00 | ........p��l�..
002C80: 40 1F 7B A8 E4 55 00 00  01 00 00 00 00 00 00 00 | @.{��U..........
002C90: F4 B9 A3 4F FD 7E 00 00  53 A0 DE 4F FD 7E 00 00 | ���O�~..S��O�~..
002CA0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002CB0: 01 00 00 00 00 00 00 00  A8 F1 1E 4F FD 7E 00 00 | ........��.O�~..
002CC0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
002CD0: EA FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ����............
002CE0: 01 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00 | ................
002CF0: 0A 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

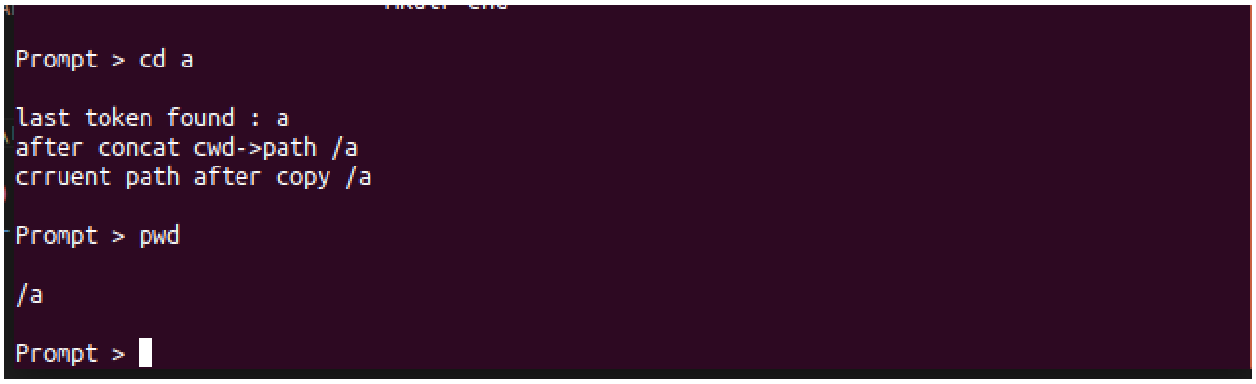


```
002D00: 00 00 00 00 00 00 00 00  F0 21 BA A6 E4 55 00 00 | ........�!���U..
002D10: FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ����............
002D20: E0 F5 1E 4F FD 7E 00 00  80 1F 00 00 FF FF 00 00 | ��.O�~..�...��..
002D30: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002D40: B0 A7 A3 4F FD 7E 00 00  48 52 DF 6C FF 7F 00 00 | ���O�~..HR�l�..
002D50: 00 00 00 00 20 00 00 00  00 00 00 00 00 00 00 00 | .... ...........
002D60: 00 00 00 00 00 00 00 00  40 50 A3 4F FD 7E 00 00 | ........@P�O�~..
002D70: 08 00 00 00 00 00 00 00  F0 21 BA A6 E4 55 00 00 | ........�!���U..
002D80: B0 5C 7C A8 E4 55 00 00  00 00 00 00 00 00 00 00 | �\|��U..........
002D90: 20 00 DE 6C FF 7F 00 00  53 A0 DE 4F FD 7E 00 00 |  .�l�..S��O�~..
002DA0: 01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002DB0: E0 C2 81 4F FD 7E 00 00  00 00 00 14 00 00 00 00 | �[illegible]O�~..........
002DC0: 80 69 5E 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | �i^O�~..........
002DD0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002DE0: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ����....����....
002DF0: 80 FD DD 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ���l�..........

002E00: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002E10: 78 FD DD 6C FF 7F 00 00  E8 61 7C A8 E4 55 00 00 | x��l�..�a|��U..
002E20: C0 FD DD 6C FF 7F 00 00  F8 FC DD 6C FF 7F 00 00 | ���l�..���l�..
002E30: F0 FC DD 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ���l�..........
002E40: D0 FD DD 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ���l�..........
002E50: 00 00 00 00 00 00 00 00  00 AC A9 20 37 60 8E 35 | .........�� 7`�5
002E60: C8 FD DD 6C FF 7F 00 00  E8 61 7C A8 E4 55 00 00 | ���l�..�a|��U..
002E70: 10 FE DD 6C FF 7F 00 00  48 FD DD 6C FF 7F 00 00 | .��l�..H��l�..
002E80: 40 FD DD 6C FF 7F 00 00  50 61 7C A8 E4 55 00 00 | @��l�..Pa|��U..
002E90: 00 00 00 00 00 00 00 00  D3 07 2A 4F FD 7E 00 00 | ........�.*O�~..
002EA0: 00 00 00 00 00 00 00 00  00 AC A9 20 37 60 8E 35 | .........�� 7`�5
002EB0: CC 3B 7C A8 E4 55 00 00  6C FE DD 6C FF 7F 31 36 | �;|��U..l��l�16
002EC0: 36 39 39 35 35 32 30 36  00 AC A9 20 37 60 8E 35 | 69955206.�� 7`�5
002ED0: C0 FD DD 6C FF 7F 00 00  C0 FD DD 6C FF 7F 00 00 | ���l�..���l�..
002EE0: 20 00 DE 6C FF 7F 00 00  45 BA 82 4F 00 7E 00 00 |  .�l�..E��O.~..
002EF0: 40 FF DD 6C FF 7F 00 00  01 00 00 00 00 00 00 00 | @��l�..........

002F00: FF FF FF FF 00 00 00 00  FF FF FF FF 00 7E 00 00 | ����....����.~..
002F10: 01 80 AD FB FF 7F 00 00  20 00 DE 6C FF 7F 00 00 | .����.. .�l�..
002F20: 20 00 DE 6C FF 7F 00 00  20 00 DE 6C FF 7F 00 00 |  .�l�.. .�l�..
002F30: 20 00 DE 6C FF 7F 00 00  2B 00 DE 6C FF 7F 00 00 |  .�l�..+.�l�..
002F40: 5E 00 DE 6C FF 7F 00 00  20 00 DE 6C FF 7F 00 00 | ^.�l�.. .�l�..
002F50: 5E 00 DE 6C FF 7F 00 00  00 00 00 00 00 00 00 00 | ^.�l�..........
002F60: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002F70: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002F80: 00 00 00 00 04 00 00 00  00 00 00 00 00 00 00 00 | ................
002F90: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002FA0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
002FB0: FF FF FF FF FF FF FF FF  00 00 00 00 00 00 00 00 | ��������........
002FC0: 00 00 00 00 00 00 00 00  00 FF DD 6C FF 7F 00 00 | .........��l�..
002FD0: FF FF FF FF FD 7E 00 00  00 00 00 04 00 00 00 00 | �����~..........
002FE0: 80 69 5E 4F FD 7E 00 00  A0 AF 5C 4F FD 7E 00 00 | �i^O�~..��\O�~..
002FF0: 00 00 00 00 00 00 00 00  00 00 00 04 00 00 00 00 | ................
```



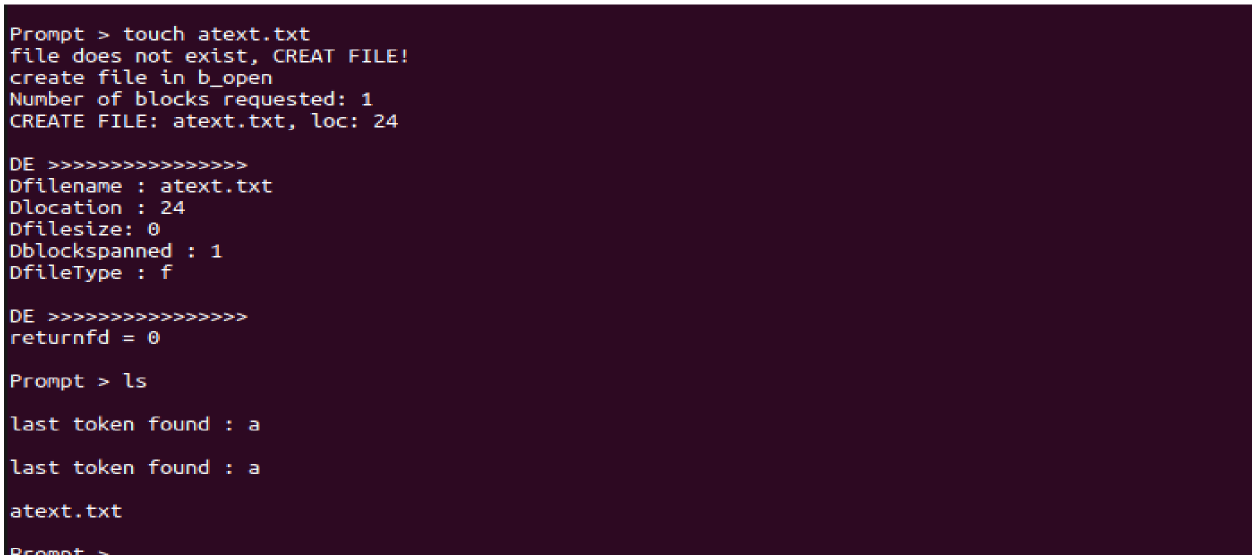
```
003000: 80 69 5E 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | �i^O�~..........
003010: 62 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | b...............
003020: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ����....����....
003030: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003040: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003050: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003060: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003070: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003080: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003090: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

003100: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003110: 3A 00 00 00 00 00 00 00  00 AC A9 20 37 60 8E 35 | :........�� 7`�5
003120: 68 F6 99 A6 E4 55 00 00  00 00 00 00 00 00 00 00 | h����U..........
003130: 60 75 99 A6 E4 55 00 00  E0 23 BA A6 E4 55 00 00 | `u���U..�#���U..
003140: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003150: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003160: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003170: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003180: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003190: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$ =
```

cmd_cd and cmd_pwd:



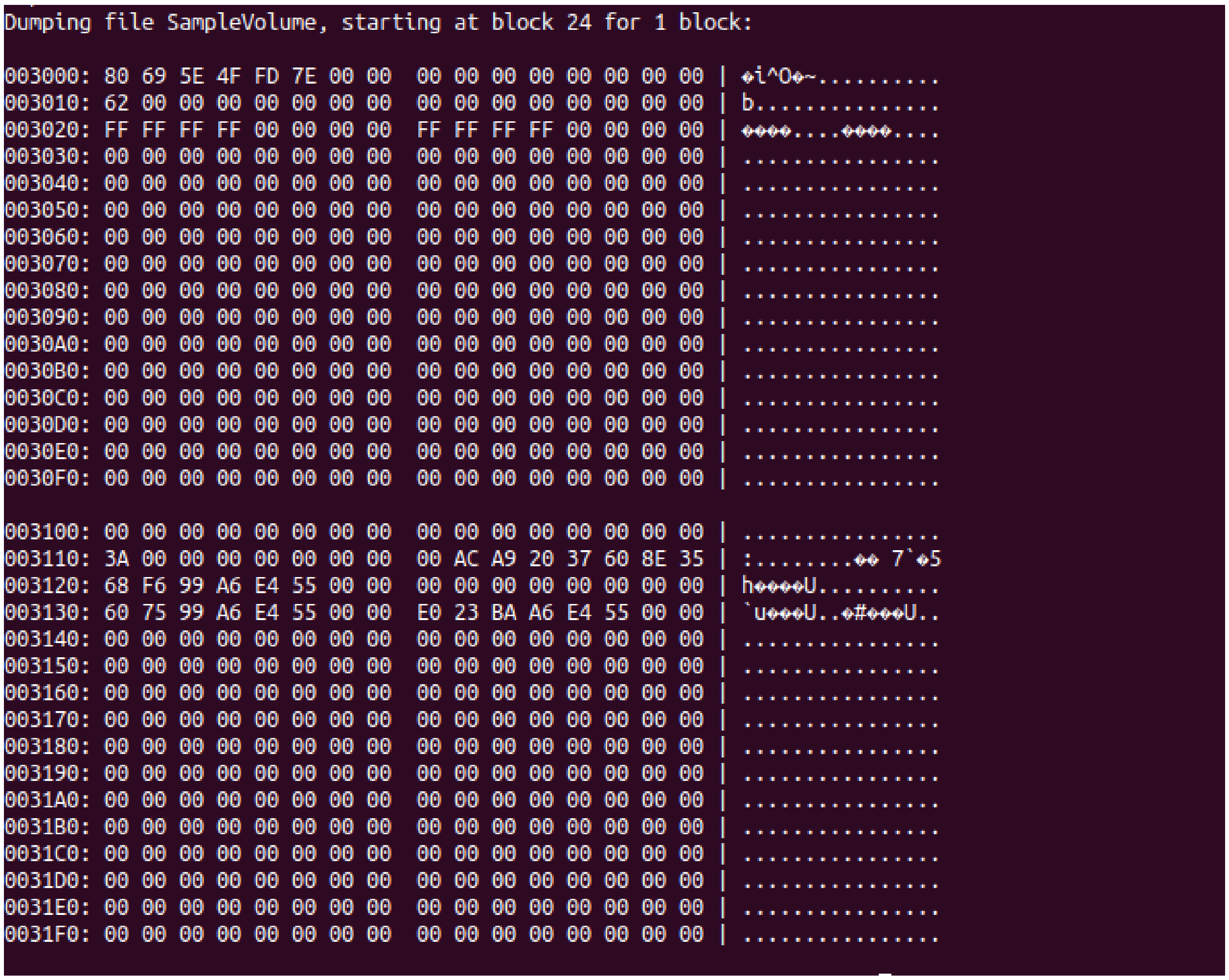
```
Prompt > cd a

last token found : a
after concat cwd->path /a
crruent path after copy /a

Prompt > pwd

/a

Prompt > 
```

Cmd_touch:



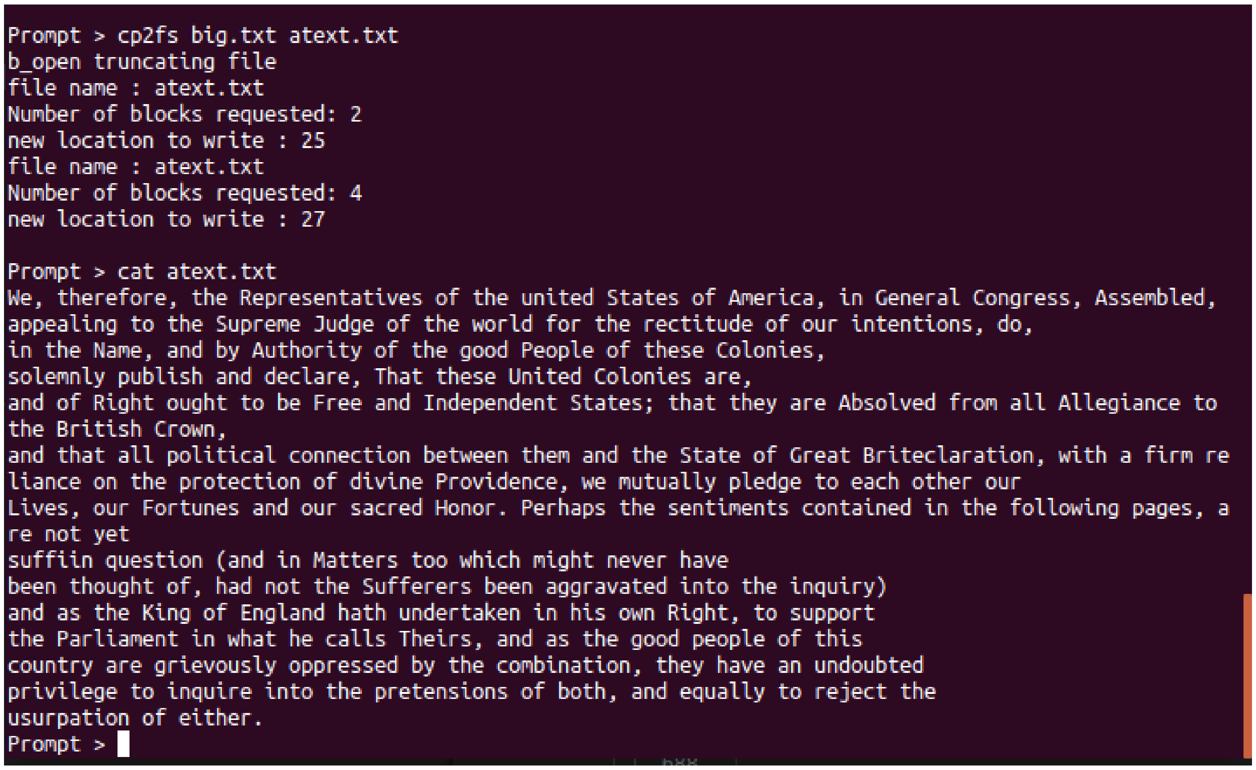
```
Prompt > touch atext.txt
file does not exist, CREAT FILE!
create file in b_open
Number of blocks requested: 1
CREATE FILE: atext.txt, loc: 24

DE >>>>>>>>>>>>>>>>
Dfilename : atext.txt
Dlocation : 24
Dfilesize: 0
Dblockspanned : 1
DfileType : f

DE >>>>>>>>>>>>>>>>
returnfd = 0

Prompt > ls

last token found : a

last token found : a

atext.txt

Prompt >
```

atext.txt hexdump:



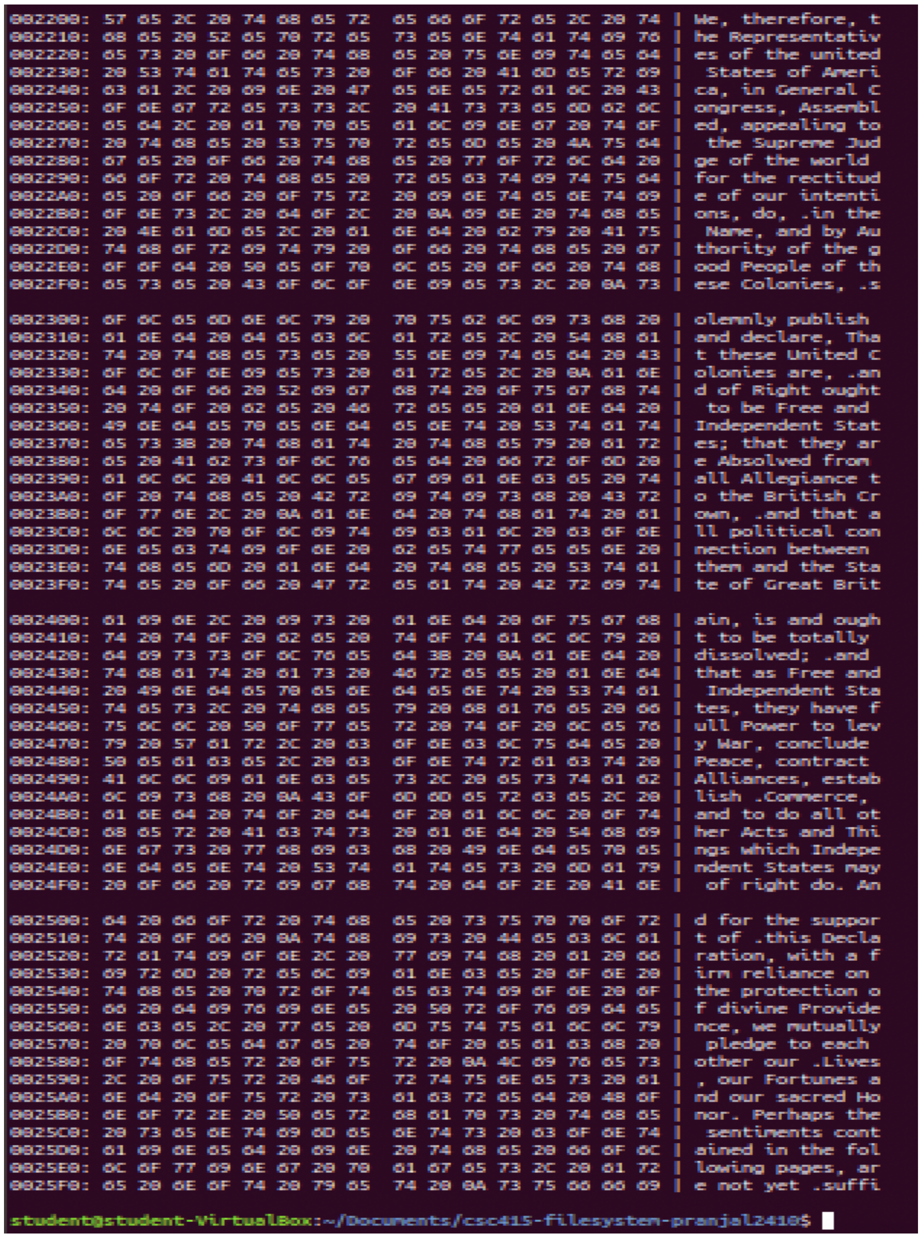
```
Dumping file SampleVolume, starting at block 24 for 1 block:

003000: 80 69 5E 4F FD 7E 00 00  00 00 00 00 00 00 00 00 | ◆i^O◆~..........
003010: 62 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | b...............
003020: FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00 | ◆◆◆◆....◆◆◆◆....
003030: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003040: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003050: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003060: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003070: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003080: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003090: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0030F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

003100: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003110: 3A 00 00 00 00 00 00 00  00 AC A9 20 37 60 8E 35 | :........◆◆ 7`◆5
003120: 68 F6 99 A6 E4 55 00 00  00 00 00 00 00 00 00 00 | h◆◆◆◆U..........
003130: 60 75 99 A6 E4 55 00 00  E0 23 BA A6 E4 55 00 00 | `u◆◆◆U..◆#◆◆◆U..
003140: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003150: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003160: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003170: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003180: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003190: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

cmd_cp2fs:



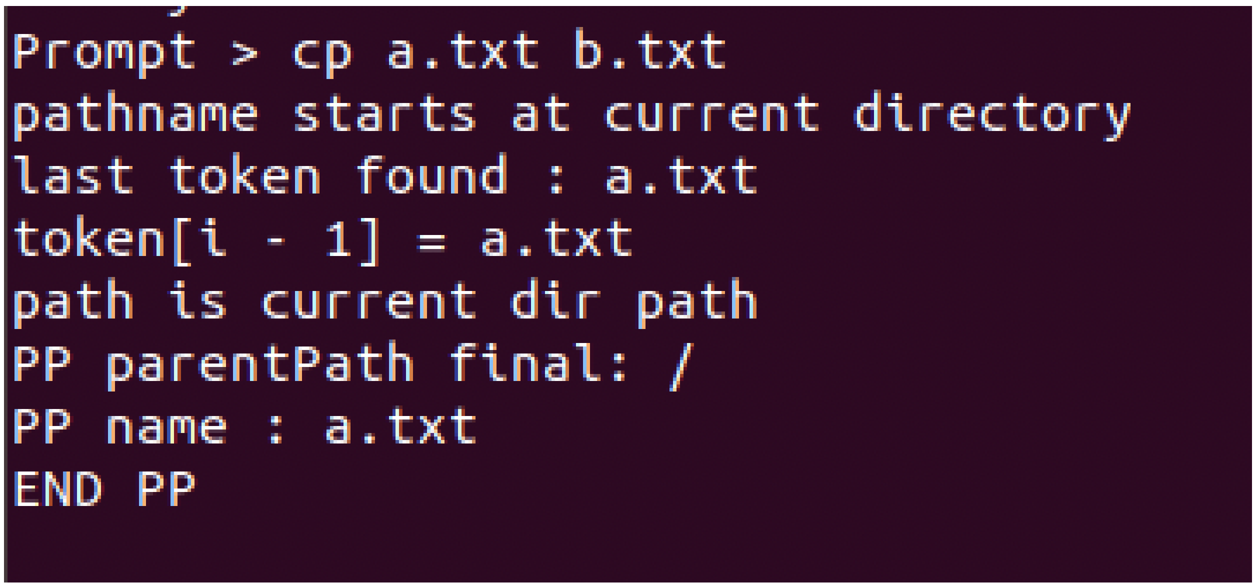
```
Prompt > cp2fs big.txt atext.txt
b_open truncating file
file name : atext.txt
Number of blocks requested: 2
new location to write : 25
file name : atext.txt
Number of blocks requested: 4
new location to write : 27

Prompt > cat atext.txt
We, therefore, the Representatives of the united States of America, in General Congress, Assembled,
appealing to the Supreme Judge of the world for the rectitude of our intentions, do,
in the Name, and by Authority of the good People of these Colonies,
solemnly publish and declare, That these United Colonies are,
and of Right ought to be Free and Independent States; that they are Absolved from all Allegiance to
the British Crown,
and that all political connection between them and the State of Great Briteclaration, with a firm re
liance on the protection of divine Providence, we mutually pledge to each other our
Lives, our Fortunes and our sacred Honor. Perhaps the sentiments contained in the following pages, a
re not yet
suffiin question (and in Matters too which might never have
been thought of, had not the Sufferers been aggravated into the inquiry)
and as the King of England hath undertaken in his own Right, to support
the Parliament in what he calls Theirs, and as the good people of this
country are grievously oppressed by the combination, they have an undoubted
privilege to inquire into the pretensions of both, and equally to reject the
usurpation of either.
Prompt >
```

cp2fs hexdump:



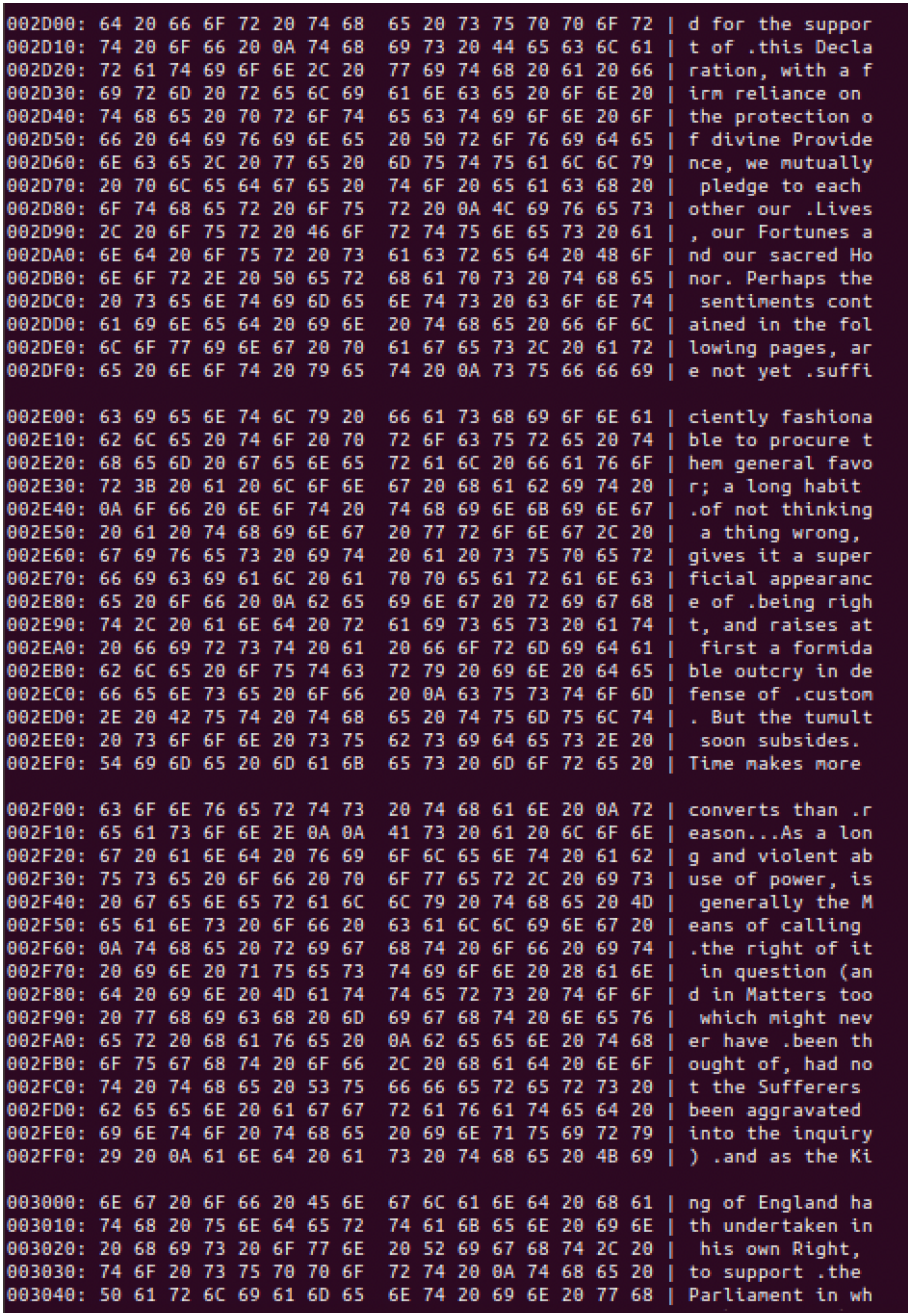
```
002200: 57 65 2C 20 74 68 65 72  65 66 6F 72 65 2C 20 74 | We, therefore, t
002210: 68 65 20 52 65 70 72 65  73 65 6E 74 61 74 69 76 | he Representativ
002220: 65 73 20 6F 66 20 74 68  65 20 75 6E 69 74 65 64 | es of the united
002230: 20 53 74 61 74 65 73 20  6F 66 20 41 6D 65 72 69 |  States of Ameri
002240: 63 61 2C 20 69 6E 20 47  65 6E 65 72 61 6C 20 43 | ca, in General C
002250: 6F 6E 67 72 65 73 73 2C  20 41 73 73 65 6D 62 6C | ongress, Assembl
002260: 65 64 2C 20 61 70 70 65  61 6C 69 6E 67 20 74 6F | ed, appealing to
002270: 20 74 68 65 20 53 75 70  72 65 6D 65 20 4A 75 64 |  the Supreme Jud
002280: 67 65 20 6F 66 20 74 68  65 20 77 6F 72 6C 64 20 | ge of the world 
002290: 66 6F 72 20 74 68 65 20  72 65 63 74 69 74 75 64 | for the rectitud
0022A0: 65 20 6F 66 20 6F 75 72  20 69 6E 74 65 6E 74 69 | e of our intenti
0022B0: 6F 6E 73 2C 20 64 6F 2C  20 0A 69 6E 20 74 68 65 | ons, do, .in the
0022C0: 20 4E 61 6D 65 2C 20 61  6E 64 20 62 79 20 41 75 |  Name, and by Au
0022D0: 74 68 6F 72 69 74 79 20  6F 66 20 74 68 65 20 67 | thority of the g
0022E0: 6F 6F 64 20 50 65 6F 70  6C 65 20 6F 66 20 74 68 | ood People of th
0022F0: 65 73 65 20 43 6F 6C 6F  6E 69 65 73 2C 20 0A 73 | ese Colonies, .s

002300: 6F 6C 65 6D 6E 6C 79 20  70 75 62 6C 69 73 68 20 | olemnly publish 
002310: 61 6E 64 20 64 65 63 6C  61 72 65 2C 20 54 68 61 | and declare, Tha
002320: 74 20 74 68 65 73 65 20  55 6E 69 74 65 64 20 43 | t these United C
002330: 6F 6C 6F 6E 69 65 73 20  61 72 65 2C 20 0A 61 6E | olonies are, .an
002340: 64 20 6F 66 20 52 69 67  68 74 20 6F 75 67 68 74 | d of Right ought
002350: 20 74 6F 20 62 65 20 46  72 65 65 20 61 6E 64 20 |  to be Free and 
002360: 49 6E 64 65 70 65 6E 64  65 6E 74 20 53 74 61 74 | Independent Stat
002370: 65 73 3B 20 74 68 61 74  20 74 68 65 79 20 61 72 | es; that they ar
002380: 65 20 41 62 73 6F 6C 76  65 64 20 66 72 6F 6D 20 | e Absolved from 
002390: 61 6C 6C 20 41 6C 6C 65  67 69 61 6E 63 65 20 74 | all Allegiance t
0023A0: 6F 20 74 68 65 20 42 72  69 74 69 73 68 20 43 72 | o the British Cr
0023B0: 6F 77 6E 2C 20 0A 61 6E  64 20 74 68 61 74 20 61 | own, .and that a
0023C0: 6C 6C 20 70 6F 6C 69 74  69 63 61 6C 20 63 6F 6E | ll political con
0023D0: 6E 65 63 74 69 6F 6E 20  62 65 74 77 65 65 6E 20 | nection between 
0023E0: 74 68 65 6D 20 61 6E 64  20 74 68 65 20 53 74 61 | them and the Sta
0023F0: 74 65 20 6F 66 20 47 72  65 61 74 20 42 72 69 74 | te of Great Brit

002400: 61 69 6E 2C 20 69 73 20  61 6E 64 20 6F 75 67 68 | ain, is and ough
002410: 74 20 74 6F 20 62 65 20  74 6F 74 61 6C 6C 79 20 | t to be totally 
002420: 64 69 73 73 6F 6C 76 65  64 3B 20 0A 61 6E 64 20 | dissolved; .and 
002430: 74 68 61 74 20 61 73 20  46 72 65 65 20 61 6E 64 | that as Free and
002440: 20 49 6E 64 65 70 65 6E  64 65 6E 74 20 53 74 61 |  Independent Sta
002450: 74 65 73 2C 20 74 68 65  79 20 68 61 76 65 20 66 | tes, they have f
002460: 75 6C 6C 20 50 6F 77 65  72 20 74 6F 20 6C 65 76 | ull Power to lev
002470: 79 20 57 61 72 2C 20 63  6F 6E 63 6C 75 64 65 20 | y War, conclude 
002480: 50 65 61 63 65 2C 20 63  6F 6E 74 72 61 63 74 20 | Peace, contract 
002490: 41 6C 6C 69 61 6E 63 65  73 2C 20 65 73 74 61 62 | Alliances, estab
0024A0: 6C 69 73 68 20 0A 43 6F  6D 6D 65 72 63 65 2C 20 | lish .Commerce, 
0024B0: 61 6E 64 20 74 6F 20 64  6F 20 61 6C 6C 20 6F 74 | and to do all ot
0024C0: 68 65 72 20 41 63 74 73  20 61 6E 64 20 54 68 69 | her Acts and Thi
0024D0: 6E 67 73 20 77 68 69 63  68 20 49 6E 64 65 70 65 | ngs which Indepe
0024E0: 6E 64 65 6E 74 20 53 74  61 74 65 73 20 6D 61 79 | ndent States may
0024F0: 20 6F 66 20 72 69 67 68  74 20 64 6F 2E 20 41 6E |  of right do. An

002500: 64 20 66 6F 72 20 74 68  65 20 73 75 70 70 6F 72 | d for the suppor
002510: 74 20 6F 66 20 0A 74 68  69 73 20 44 65 63 6C 61 | t of .this Decla
002520: 72 61 74 69 6F 6E 2C 20  77 69 74 68 20 61 20 66 | ration, with a f
002530: 69 72 6D 20 72 65 6C 69  61 6E 63 65 20 6F 6E 20 | irm reliance on 
002540: 74 68 65 20 70 72 6F 74  65 63 74 69 6F 6E 20 6F | the protection o
002550: 66 20 64 69 76 69 6E 65  20 50 72 6F 76 69 64 65 | f divine Provide
002560: 6E 63 65 2C 20 77 65 20  6D 75 74 75 61 6C 6C 79 | nce, we mutually
002570: 20 70 6C 65 64 67 65 20  74 6F 20 65 61 63 68 20 |  pledge to each 
002580: 6F 74 68 65 72 20 6F 75  72 20 0A 4C 69 76 65 73 | other our .Lives
002590: 2C 20 6F 75 72 20 46 6F  72 74 75 6E 65 73 20 61 | , our Fortunes a
0025A0: 6E 64 20 6F 75 72 20 73  61 63 72 65 64 20 48 6F | nd our sacred Ho
0025B0: 6E 6F 72 2E 20 50 65 72  68 61 70 73 20 74 68 65 | nor. Perhaps the
0025C0: 20 73 65 6E 74 69 6D 65  6E 74 73 20 63 6F 6E 74 |  sentiments cont
0025D0: 61 69 6E 65 64 20 69 6E  20 74 68 65 20 66 6F 6C | ained in the fol
0025E0: 6C 6F 77 69 6E 67 20 70  61 67 65 73 2C 20 61 72 | lowing pages, ar
0025F0: 65 20 6E 6F 74 20 79 65  74 20 0A 73 75 66 66 69 | e not yet .suffi

student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$ 
```

Cmd_cp:

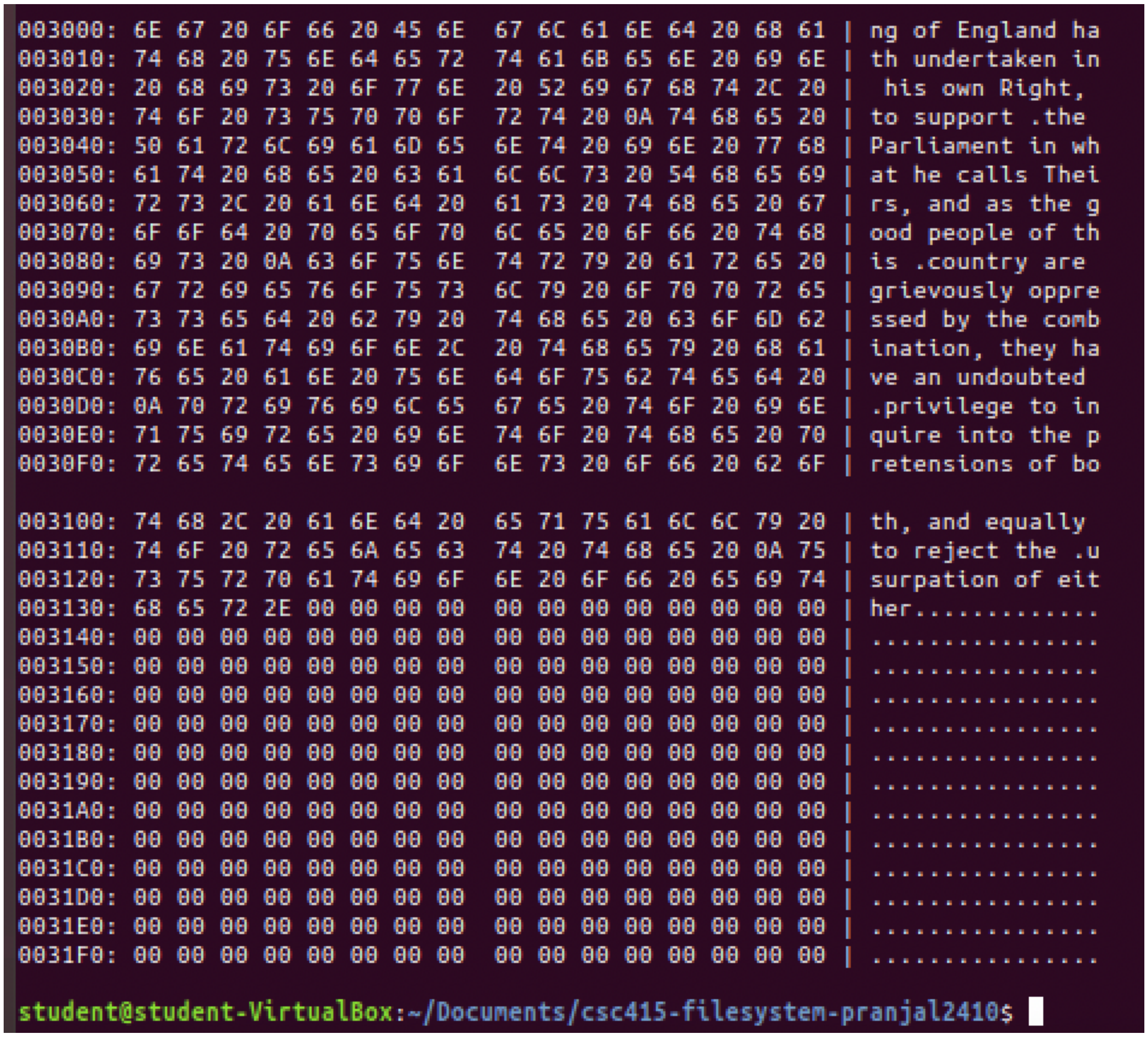


```
Prompt > cp a.txt b.txt
pathname starts at current directory
last token found : a.txt
token[i - 1] = a.txt
path is current dir path
PP parentPath final: /
PP name : a.txt
END PP
```

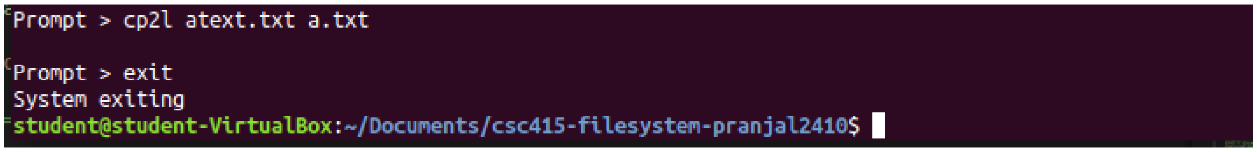
cp hexdump:



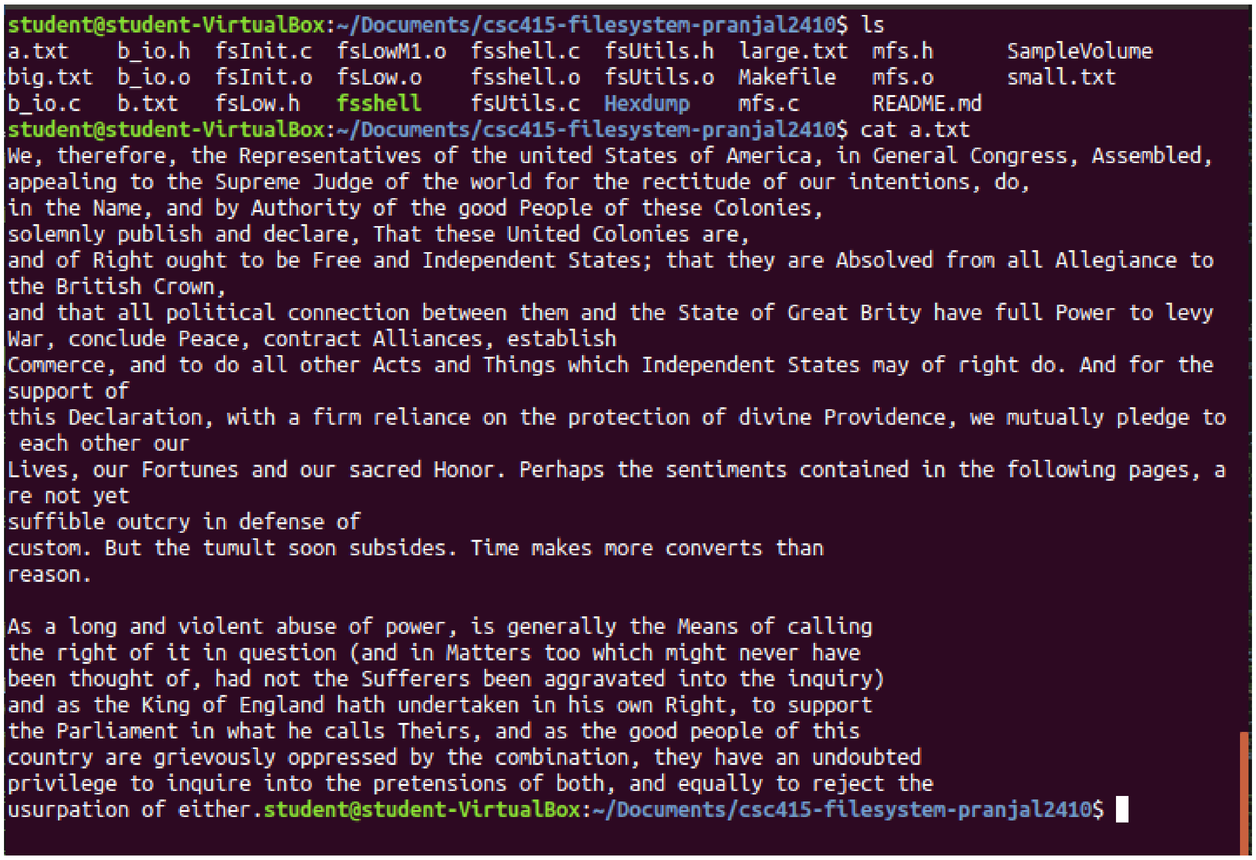
```
002D00: 64 20 66 6F 72 20 74 68  65 20 73 75 70 70 6F 72 | d for the suppor
002D10: 74 20 6F 66 20 0A 74 68  69 73 20 44 65 63 6C 61 | t of .this Decla
002D20: 72 61 74 69 6F 6E 2C 20  77 69 74 68 20 61 20 66 | ration, with a f
002D30: 69 72 6D 20 72 65 6C 69  61 6E 63 65 20 6F 6E 20 | irm reliance on 
002D40: 74 68 65 20 70 72 6F 74  65 63 74 69 6F 6E 20 6F | the protection o
002D50: 66 20 64 69 76 69 6E 65  20 50 72 6F 76 69 64 65 | f divine Provide
002D60: 6E 63 65 2C 20 77 65 20  6D 75 74 75 61 6C 6C 79 | nce, we mutually
002D70: 20 70 6C 65 64 67 65 20  74 6F 20 65 61 63 68 20 |  pledge to each 
002D80: 6F 74 68 65 72 20 6F 75  72 20 0A 4C 69 76 65 73 | other our .Lives
002D90: 2C 20 6F 75 72 20 46 6F  72 74 75 6E 65 73 20 61 | , our Fortunes a
002DA0: 6E 64 20 6F 75 72 20 73  61 63 72 65 64 20 48 6F | nd our sacred Ho
002DB0: 6E 6F 72 2E 20 50 65 72  68 61 70 73 20 74 68 65 | nor. Perhaps the
002DC0: 20 73 65 6E 74 69 6D 65  6E 74 73 20 63 6F 6E 74 |  sentiments cont
002DD0: 61 69 6E 65 64 20 69 6E  20 74 68 65 20 66 6F 6C | ained in the fol
002DE0: 6C 6F 77 69 6E 67 20 70  61 67 65 73 2C 20 61 72 | lowing pages, ar
002DF0: 65 20 6E 6F 74 20 79 65  74 20 0A 73 75 66 66 69 | e not yet .suffi

002E00: 63 69 65 6E 74 6C 79 20  66 61 73 68 69 6F 6E 61 | ciently fashiona
002E10: 62 6C 65 20 74 6F 20 70  72 6F 63 75 72 65 20 74 | ble to procure t
002E20: 68 65 6D 20 67 65 6E 65  72 61 6C 20 66 61 76 6F | hem general favo
002E30: 72 3B 20 61 20 6C 6F 6E  67 20 68 61 62 69 74 20 | r; a long habit 
002E40: 0A 6F 66 20 6E 6F 74 20  74 68 69 6E 6B 69 6E 67 | .of not thinking
002E50: 20 61 20 74 68 69 6E 67  20 77 72 6F 6E 67 2C 20 |  a thing wrong, 
002E60: 67 69 76 65 73 20 69 74  20 61 20 73 75 70 65 72 | gives it a super
002E70: 66 69 63 69 61 6C 20 61  70 70 65 61 72 61 6E 63 | ficial appearanc
002E80: 65 20 6F 66 20 0A 62 65  69 6E 67 20 72 69 67 68 | e of .being righ
002E90: 74 2C 20 61 6E 64 20 72  61 69 73 65 73 20 61 74 | t, and raises at
002EA0: 20 66 69 72 73 74 20 61  20 66 6F 72 6D 69 64 61 |  first a formida
002EB0: 62 6C 65 20 6F 75 74 63  72 79 20 69 6E 20 64 65 | ble outcry in de
002EC0: 66 65 6E 73 65 20 6F 66  20 0A 63 75 73 74 6F 6D | fense of .custom
002ED0: 2E 20 42 75 74 20 74 68  65 20 74 75 6D 75 6C 74 | . But the tumult
002EE0: 20 73 6F 6F 6E 20 73 75  62 73 69 64 65 73 2E 20 |  soon subsides. 
002EF0: 54 69 6D 65 20 6D 61 6B  65 73 20 6D 6F 72 65 20 | Time makes more 

002F00: 63 6F 6E 76 65 72 74 73  20 74 68 61 6E 20 0A 72 | converts than .r
002F10: 65 61 73 6F 6E 2E 0A 0A  41 73 20 61 20 6C 6F 6E | eason...As a lon
002F20: 67 20 61 6E 64 20 76 69  6F 6C 65 6E 74 20 61 62 | g and violent ab
002F30: 75 73 65 20 6F 66 20 70  6F 77 65 72 2C 20 69 73 | use of power, is
002F40: 20 67 65 6E 65 72 61 6C  6C 79 20 74 68 65 20 4D |  generally the M
002F50: 65 61 6E 73 20 6F 66 20  63 61 6C 6C 69 6E 67 20 | eans of calling 
002F60: 0A 74 68 65 20 72 69 67  68 74 20 6F 66 20 69 74 | .the right of it
002F70: 20 69 6E 20 71 75 65 73  74 69 6F 6E 20 28 61 6E |  in question (an
002F80: 64 20 69 6E 20 4D 61 74  74 65 72 73 20 74 6F 6F | d in Matters too
002F90: 20 77 68 69 63 68 20 6D  69 67 68 74 20 6E 65 76 |  which might nev
002FA0: 65 72 20 68 61 76 65 20  0A 62 65 65 6E 20 74 68 | er have .been th
002FB0: 6F 75 67 68 74 20 6F 66  2C 20 68 61 64 20 6E 6F | ought of, had no
002FC0: 74 20 74 68 65 20 53 75  66 66 65 72 65 72 73 20 | t the Sufferers 
002FD0: 62 65 65 6E 20 61 67 67  72 61 76 61 74 65 64 20 | been aggravated 
002FE0: 69 6E 74 6F 20 74 68 65  20 69 6E 71 75 69 72 79 | into the inquiry
002FF0: 29 20 0A 61 6E 64 20 61  73 20 74 68 65 20 4B 69 | ) .and as the Ki

003000: 6E 67 20 6F 66 20 45 6E  67 6C 61 6E 64 20 68 61 | ng of England ha
003010: 74 68 20 75 6E 64 65 72  74 61 6B 65 6E 20 69 6E | th undertaken in
003020: 20 68 69 73 20 6F 77 6E  20 52 69 67 68 74 2C 20 |  his own Right, 
003030: 74 6F 20 73 75 70 70 6F  72 74 20 0A 74 68 65 20 | to support .the 
003040: 50 61 72 6C 69 61 6D 65  6E 74 20 69 6E 20 77 68 | Parliament in wh
```


```
003000: 6E 67 20 6F 66 20 45 6E  67 6C 61 6E 64 20 68 61 | ng of England ha
003010: 74 68 20 75 6E 64 65 72  74 61 6B 65 6E 20 69 6E | th undertaken in
003020: 20 68 69 73 20 6F 77 6E  20 52 69 67 68 74 2C 20 |  his own Right,
003030: 74 6F 20 73 75 70 70 6F  72 74 20 0A 74 68 65 20 | to support .the
003040: 50 61 72 6C 69 61 6D 65  6E 74 20 69 6E 20 77 68 | Parliament in wh
003050: 61 74 20 68 65 20 63 61  6C 6C 73 20 54 68 65 69 | at he calls Thei
003060: 72 73 2C 20 61 6E 64 20  61 73 20 74 68 65 20 67 | rs, and as the g
003070: 6F 6F 64 20 70 65 6F 70  6C 65 20 6F 66 20 74 68 | ood people of th
003080: 69 73 20 0A 63 6F 75 6E  74 72 79 20 61 72 65 20 | is .country are
003090: 67 72 69 65 76 6F 75 73  6C 79 20 6F 70 70 72 65 | grievously oppre
0030A0: 73 73 65 64 20 62 79 20  74 68 65 20 63 6F 6D 62 | ssed by the comb
0030B0: 69 6E 61 74 69 6F 6E 2C  20 74 68 65 79 20 68 61 | ination, they ha
0030C0: 76 65 20 61 6E 20 75 6E  64 6F 75 62 74 65 64 20 | ve an undoubted
0030D0: 0A 70 72 69 76 69 6C 65  67 65 20 74 6F 20 69 6E | .privilege to in
0030E0: 71 75 69 72 65 20 69 6E  74 6F 20 74 68 65 20 70 | quire into the p
0030F0: 72 65 74 65 6E 73 69 6F  6E 73 20 6F 66 20 62 6F | retensions of bo

003100: 74 68 2C 20 61 6E 64 20  65 71 75 61 6C 6C 79 20 | th, and equally
003110: 74 6F 20 72 65 6A 65 63  74 20 74 68 65 20 0A 75 | to reject the .u
003120: 73 75 72 70 61 74 69 6F  6E 20 6F 66 20 65 69 74 | surpation of eit
003130: 68 65 72 2E 00 00 00 00  00 00 00 00 00 00 00 00 | her.............
003140: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003150: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003160: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003170: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003180: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
003190: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0031F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$
```

cmd_cp2l:

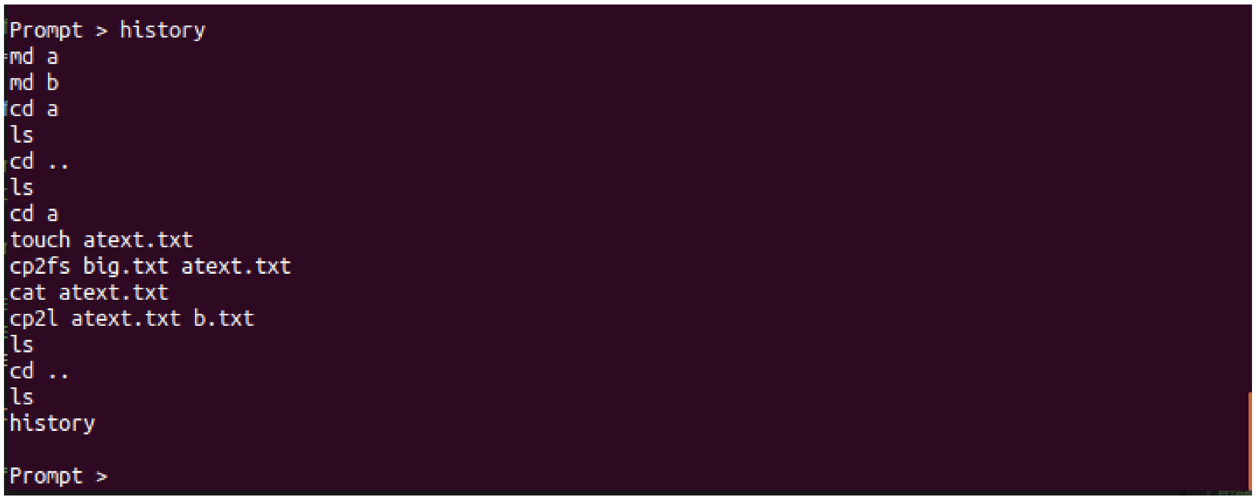

```
Prompt > cp2l atext.txt a.txt

Prompt > exit
System exiting
student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$
```

cmd_cp2l terminal:



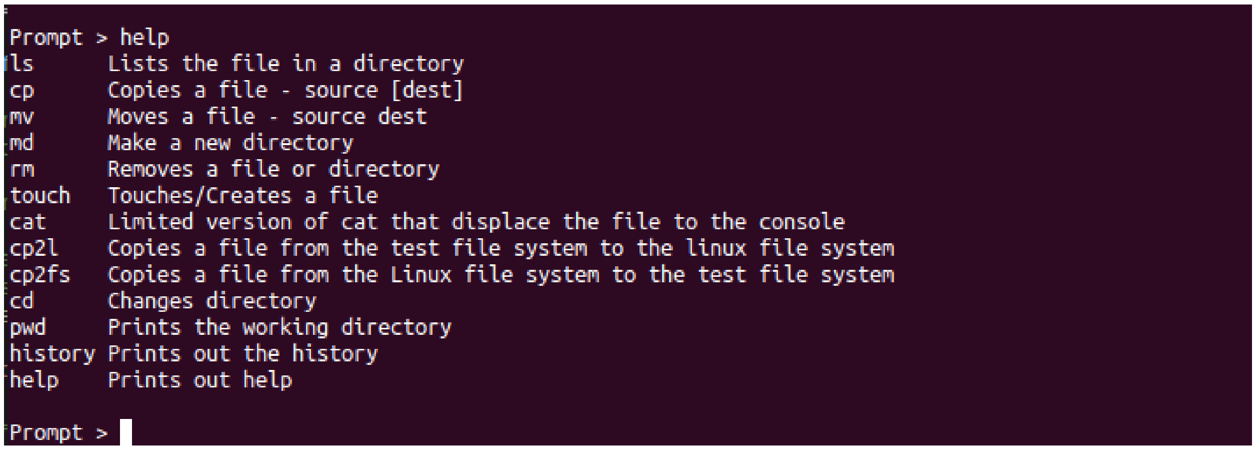
```
student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$ ls
a.txt    b_io.h  fsInit.c  fsLowM1.o  fsshell.c  fsUtils.h  large.txt  mfs.h      SampleVolume
big.txt  b_io.o  fsInit.o  fsLow.o    fsshell.o  fsUtils.o  Makefile   mfs.o      small.txt
b_io.c   b.txt   fsLow.h   fsshell    fsUtils.c  Hexdump    mfs.c      README.md
student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$ cat a.txt
We, therefore, the Representatives of the united States of America, in General Congress, Assembled,
appealing to the Supreme Judge of the world for the rectitude of our intentions, do,
in the Name, and by Authority of the good People of these Colonies,
solemnly publish and declare, That these United Colonies are,
and of Right ought to be Free and Independent States; that they are Absolved from all Allegiance to
the British Crown,
and that all political connection between them and the State of Great Brity have full Power to levy
War, conclude Peace, contract Alliances, establish
Commerce, and to do all other Acts and Things which Independent States may of right do. And for the
support of
this Declaration, with a firm reliance on the protection of divine Providence, we mutually pledge to
 each other our
Lives, our Fortunes and our sacred Honor. Perhaps the sentiments contained in the following pages, a
re not yet
suffible outcry in defense of
custom. But the tumult soon subsides. Time makes more converts than
reason.

As a long and violent abuse of power, is generally the Means of calling
the right of it in question (and in Matters too which might never have
been thought of, had not the Sufferers been aggravated into the inquiry)
and as the King of England hath undertaken in his own Right, to support
the Parliament in what he calls Theirs, and as the good people of this
country are grievously oppressed by the combination, they have an undoubted
privilege to inquire into the pretensions of both, and equally to reject the
usurpation of either.student@student-VirtualBox:~/Documents/csc415-filesystem-pranjal2410$ 
```

cmd_history:



```
Prompt > history
md a
md b
cd a
ls
cd ..
ls
cd a
touch atext.txt
cp2fs big.txt atext.txt
cat atext.txt
cp2l atext.txt b.txt
ls
cd ..
ls
history

Prompt > 
```

cmd_help:



```
Prompt > help
ls      Lists the file in a directory
cp      Copies a file - source [dest]
mv      Moves a file - source dest
md      Make a new directory
rm      Removes a file or directory
touch   Touches/Creates a file
cat     Limited version of cat that displace the file to the console
cp2l    Copies a file from the test file system to the linux file system
cp2fs   Copies a file from the Linux file system to the test file system
cd      Changes directory
pwd     Prints the working directory
history Prints out the history
help    Prints out help

Prompt > 
```