Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

File System Project

Team: Blob

Team Member	Student ID
Natalie	920698945
Westley Cho	922841683
Etinosa Osagie-Amayo	921582542
Angelo Lance Quetua	921008618
Bryan	916956188

Github:

https://github.com/CSC415-2023-Fall/csc415-filesystem-westleyc30/tree/main

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Table of contents

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Team: Blob

CSC 415-03 Operating Systems

File System

Description:

- For our file system project, what our team created was a series of main and header files that contained the functionality of common linux file system commands. Each of these commands, ls, cd, md, pwd, touch, cat, rm, cp, mv, cp2fs, and cp2l, all of them are recreated from various main and helper functions and structures, and act similar to the Linux file system. One example comes from ls, which shows all the content that was loaded onto disk like directories or files. With our file system, it is based of the FAT file system, in which we allocated a volume control block, a File Allocation Table map for the entire file system, a root directory, and eventually files that will either be contiguous or discontiguous as we keep adding or removing files over time

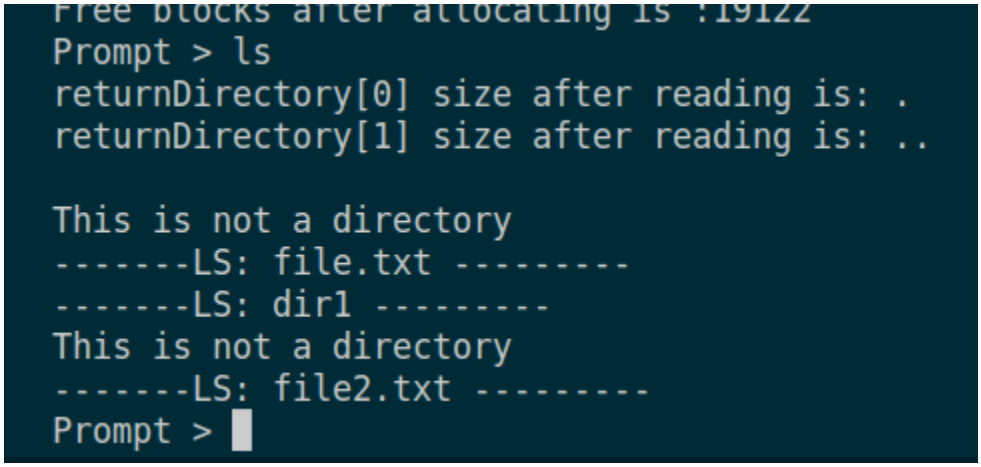
How our driver program works:

- With our driver program, it all works when using the commands that are initiated by fsshell.c, mainly each of the cmd functions, like ls, cp, mv, md, rm, touch, cat, cp2l, cp2fs, cd, and pwd. With each of our files, mfs.c, directories.c, FATInit.c, and b_io.c, they would allow each of those functions to work and deliver an output or edit certain linux files.

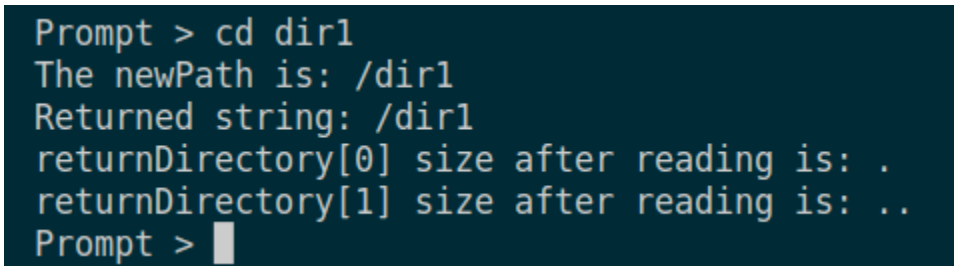
Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Screenshots/Test routines for each command

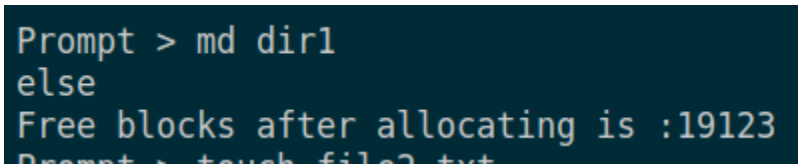
1. ls function:



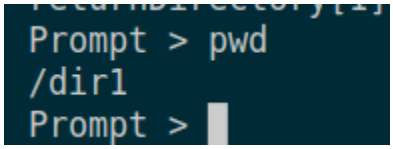
2. cd function:



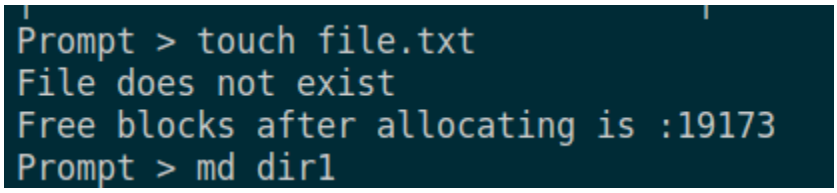
3. md function:



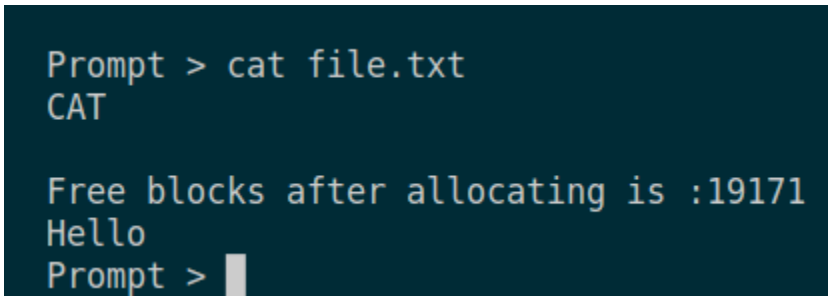
4. pwd function:



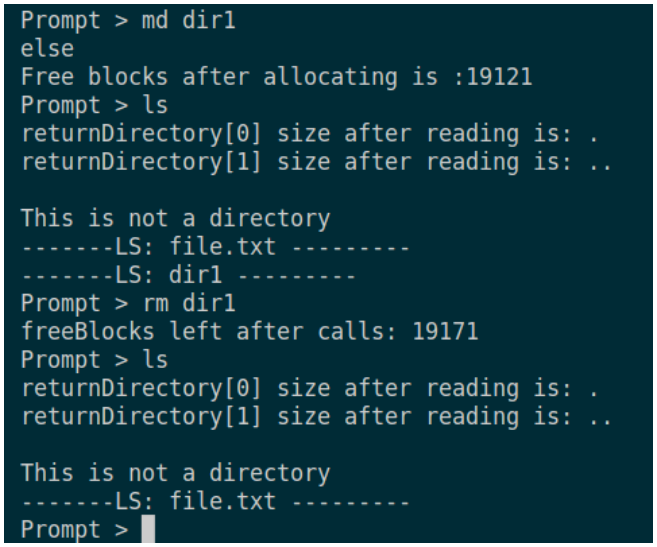
5. touch function:



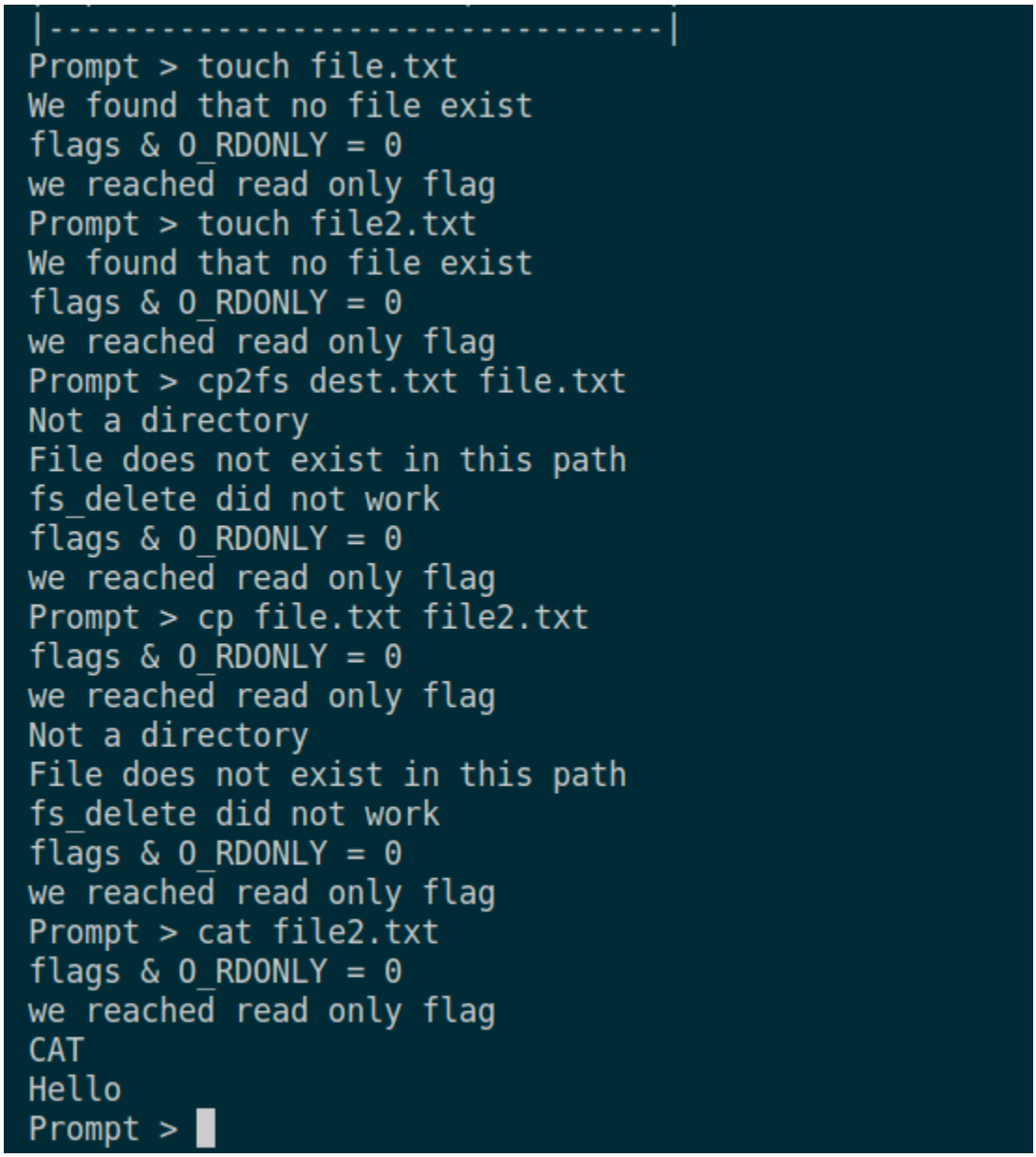
6. cat function:



7. rm function:



8. cp function:



9. mv function:

Move file into dir1

```
Prompt > ls
returnDirectory[0] size after reading is: .
returnDirectory[1] size after reading is: ..

-------LS: file ---------
-------LS: dir1 ---------
-------LS: dir2 ---------
Prompt > mv file dir1
```

After move file is gone when we check with ls

```
Prompt > ls
returnDirectory[0] size after reading is: .
returnDirectory[1] size after reading is: ..

-------LS: dir1 ---------
-------LS: dir2 ---------
Prompt >
```

We cd to dir1 where file is moved to and do an ls and we see that it was indeed moved there

```
Prompt > cd dir1
returnDirectory[0] size after reading is: .
returnDirectory[1] size after reading is: ..
Prompt > ls
returnDirectory[0] size after reading is: .
returnDirectory[1] size after reading is: ..

-------LS: file ---------
```

10. cp2fs function

Create a new file to copy contents of our linux file to

We call cat to show there is nothing in the file yet

```
Prompt > touch file
We found that no file exist
Free blocks after allocating is :19173
flags & O_RDONLY = 0
we reached read only flag
Prompt > cat file
flags & O_RDONLY = 0
we reached read only flag
CAT

Exiting while loop in b_read
Prompt >
```

Now we call cp2fs

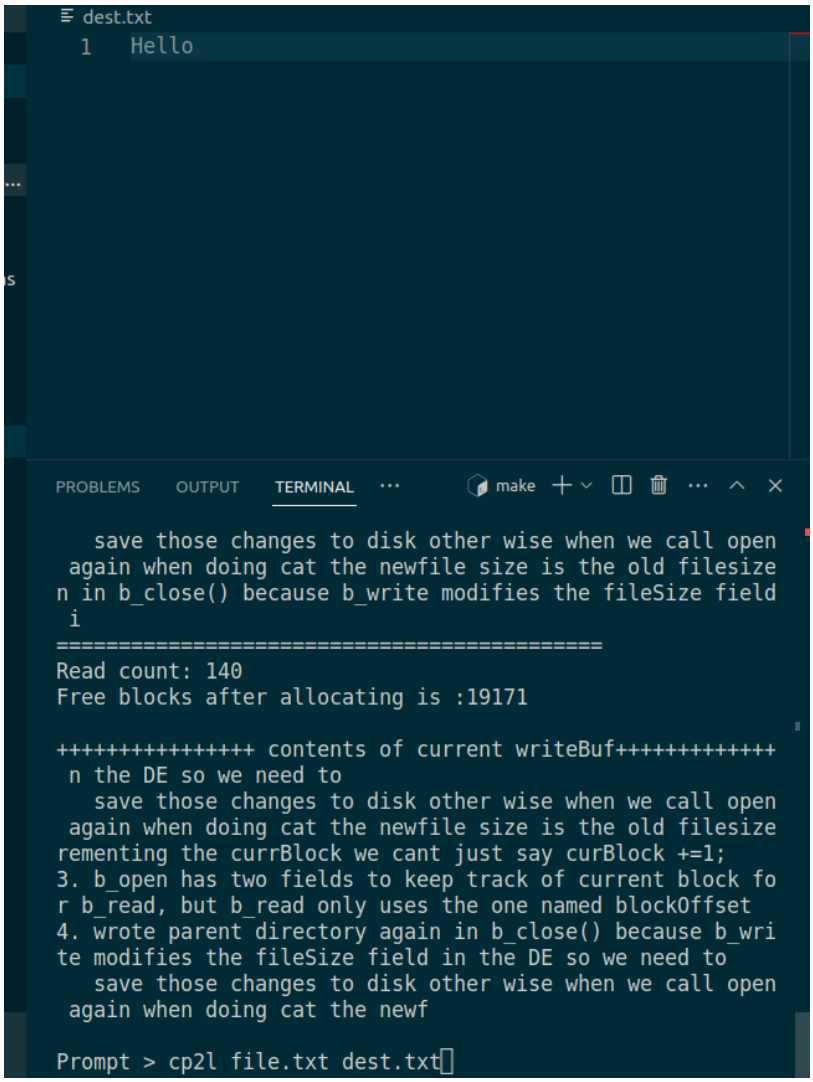
```
Prompt > cp2fs src.txt file
freeBlocks left after calls: 19174
File does not exist in this path
fs_delete did not work
Free blocks after allocating is :19173
flags & O_RDONLY = 0
we reached read only flag
Prompt >
```

Call cat command on file to check that the contents were copied:

```
Prompt > cp2fs src.txt file
freeBlocks left after calls: 19174
File does not exist in this path
fs_delete did not work
Free blocks after allocating is :19173
flags & O_RDONLY = 0
we reached read only flag
Prompt > cat file
flags & O_RDONLY = 0
we reached read only flag
CAT

Free blocks after allocating is :19171
helloooo copyyy fixes:
1. b_open RDOnly flags now get set off (we needed to add parenthesis)
2. b_read needs to call getNextBlock when incrementing the currBlock we cant jus
t say curBlock +=1;
3. b_open has two fields to keep track of current block for b_read, but b_read o
nly uses the one named blockOffset
4. wrote parent directory again in b_close() because b_write modifies the fileSi
Prompt >
```

11. cp2l function:

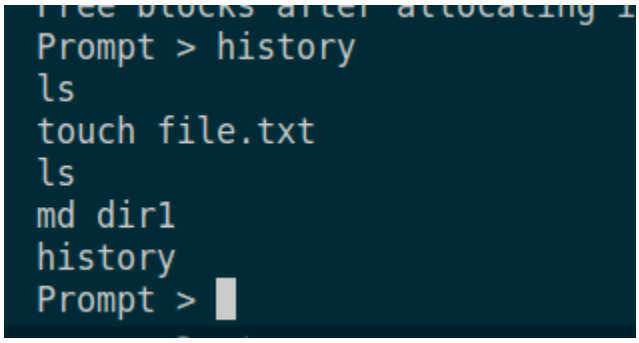
Before using cp2l:



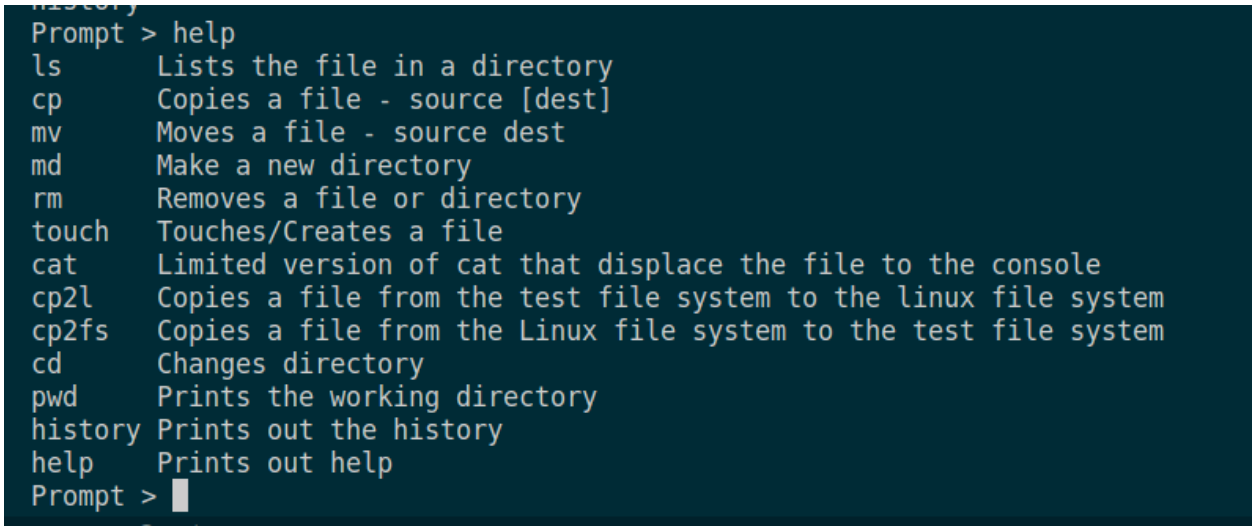
Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

After using Cp2l:

12. history:



13. help



Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Team: Blob

CSC 415-03 Operating Systems

Milestone 1

Descriptions:

1. **Volume Control Block:**
    a. For the volume control block, we initialized with a structure that contains the main information needed for the current volume, such as when the free space starts, how many free blocks there are, and what is the signature that it will hold.
2. **Free Space Management:**

    For the Free Space Management, we would use a contiguous chain for the VCB, a map for our free space, the root directory, and any leftover space that we can use to allocate any file. However, with the VCB, our map, and the root directory, we leave those blocks being used, while the leftover will be free for anything else that the user wants to allocate.
3. **Root Directory:**
    a. For the root directory, it allows us to not only initialize just the root directory (as well as the “..” directory), but also allow initialization of any directory. In addition, we also gave it various pieces of metadata, ranging from the name, location, how big it is, various date-related attributes, and if it’s a directory or not.

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

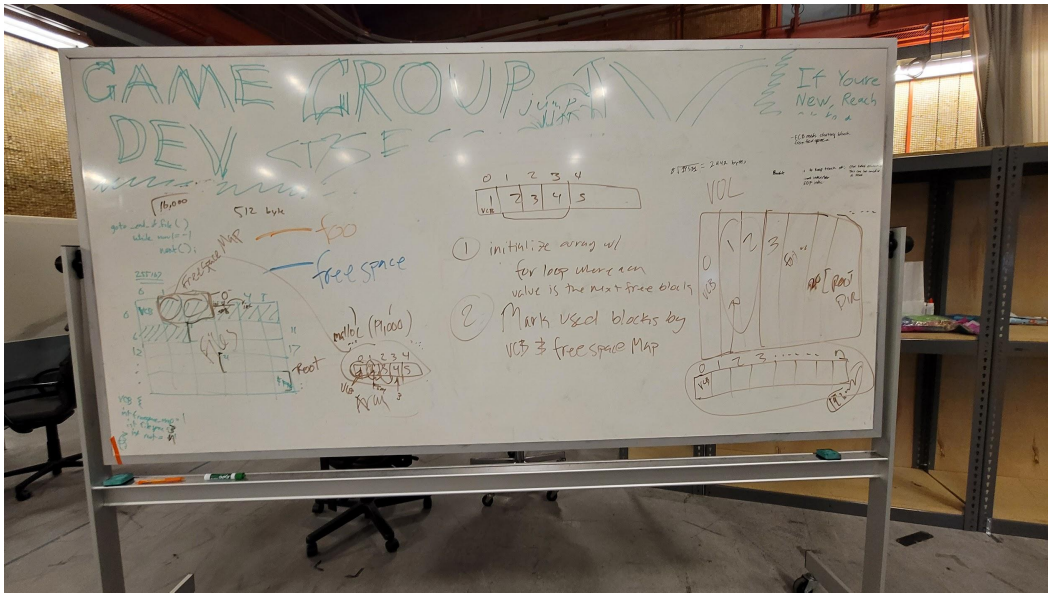
Roles:

Section	Who worked on it
Volume Control Block	**Precious**
Free Space Management	**Natalie, Angelo, and Bryan**
Root Directory	**Bryan and Westley**

Approach / What we Did:

1. The first step our group approached when doing this assignment was focusing on when to meet up at certain times over the weekend. Once we get the chance to work with each other, mainly 3:30pm on Saturday and 1pm on Sunday, we would be discussing certain roles for certain people, such as who will work on the root directory, free space management, or the volume control block. This would also include multiple people working on a certain aspect of the code, and possibly one person for one section, if they decided to work on it themselves. This also takes into account our five total members and there were three different sections of milestone one.
    a. Once we decided on our given roles, we would give time to discuss with each other on how we were going to approach our sections, while those that were doing a section by themselves would be working by themselves before we had to check in with each other.

2. The next step that our group took, while still doing an online discussion during the planning stage, was to help clear up some confusion within the milestone one step. One of the biggest concerns that we came across was being able to figure out our own free space management system. While a lot of the issues came from translating our idea into code, some of the issues came from what certain functions we were going to use, such as either using a bitmap with our FAT system, how we were going to store each of the various free blocks, etc.
    a. One of the biggest changes to approach when working as a group was instead of meeting over online, we would instead work in person so that we can easily draw out ideas near each other through white boarding, such as being able to visually see how our volumes on disk work, and how our free space management would relate to it:



3. While continuing with our approach, our group would tend to come across massive changes within how our code is structured. Two of the biggest examples in our code comes from the VCB and FreeSpace initialization, as some parts of our code would include additions or having to remake certain parts. Some of these include adding more members for our VCB,

```
typedef struct vcb {
    int blocks; // amou
    int blockSize; // si
    int freeBlocks; // a
    int fatStart; // blo
    int freeSpaceStart;
    int rootDirStart; //
    long signature; // s
} vcb;
```

```
+ typedef struct vcb {
+       int blocks;
+       int blockSize;
+       int freeBlocks;
+       int freeSpaceStart;
+       int rootDirStart;
+       long signature;
+ } vcb;
```

or having to utilize a struct for our freespace map instead of a regular array.

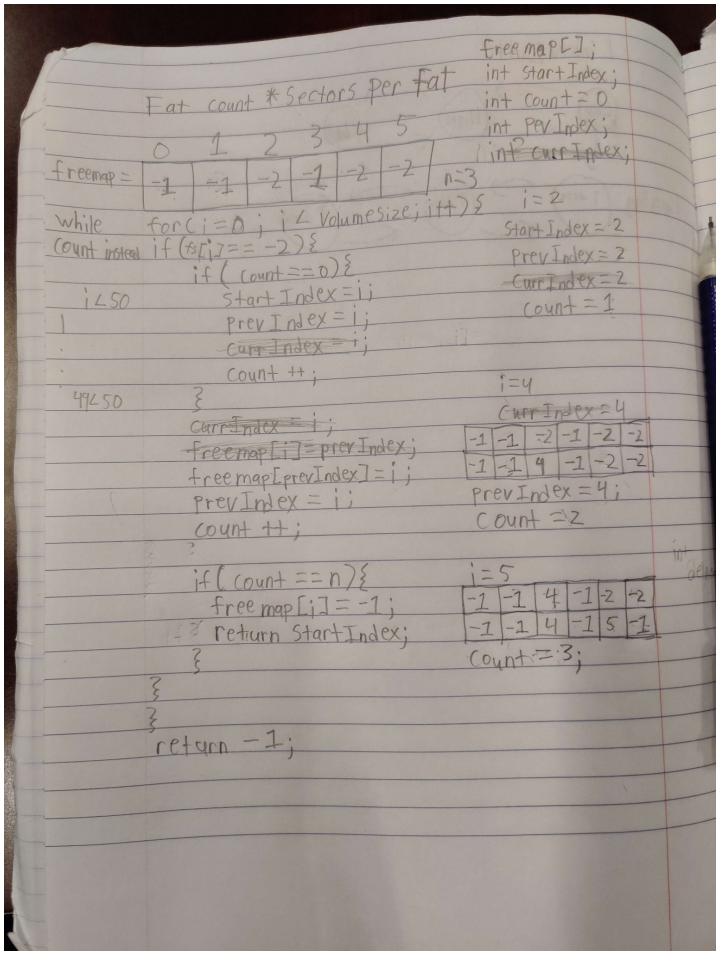
```
typedef struct blockIndicator {
    int nextBlock;
    int usedBlock; // 0 for unused, 1 for used
}blockIndicator; // size is 8 bytes
```

```
int FATMap[] = malloc(FATSize);
```

Overtime, many of these changes would end up becoming a part of the final product for our code.

4. Currently, our freespace allocates blocks contiguously but there will come a point where our memory becomes fragmented due to adding, deleting , appending, and etc which contiguous allocation wouldn't account for well.

In the photo above, we went over a case where it is possible that a smaller file gets "deleted", the blocks get freed, and a new bigger file wants the space of the smaller file plus an additional block. With our current allocation method, we have no way of linking block 4 to block 15 meaning we wouldn't be able to allocate enough space for the bigger file. This is where this pseudo-algorithm comes in.



The algorithm works by searching our FAT for one free block at a time and linking the current free block found to the previously found free block up to the requested amount of blocks. If not enough blocks were found and linked, we know we don't have enough space to accommodate the file and we return -1 to indicate so. We plan to revise this algorithm and implement it later on.

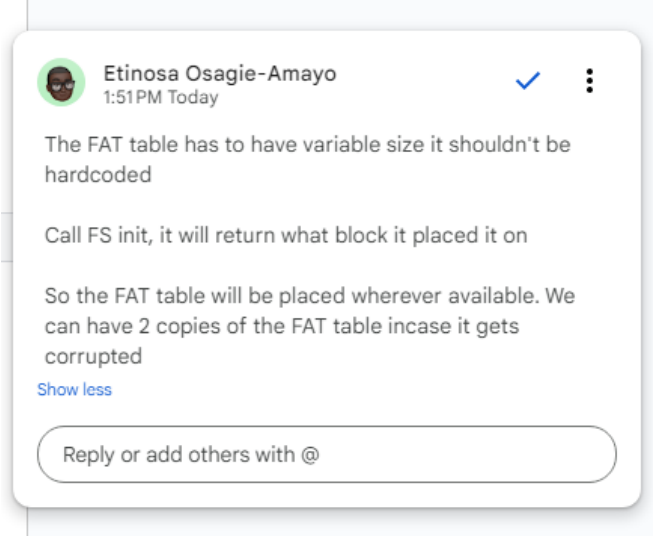
5. For the final step, once our team managed to gain a good understanding of how our logic would work, as well as a few more trips to office hours for check ups for said logic, our team would be coding each piece in each of its own files. However, while we do code each of the given logic, sometimes some members might have massive changes as well that would make it easier later on to work with. Once we have finished inputting our logic, each of us would go through each hex

dump, translate certain addresses to figure out which belongs to one of the three sections, and have it as an analysis in our write up.

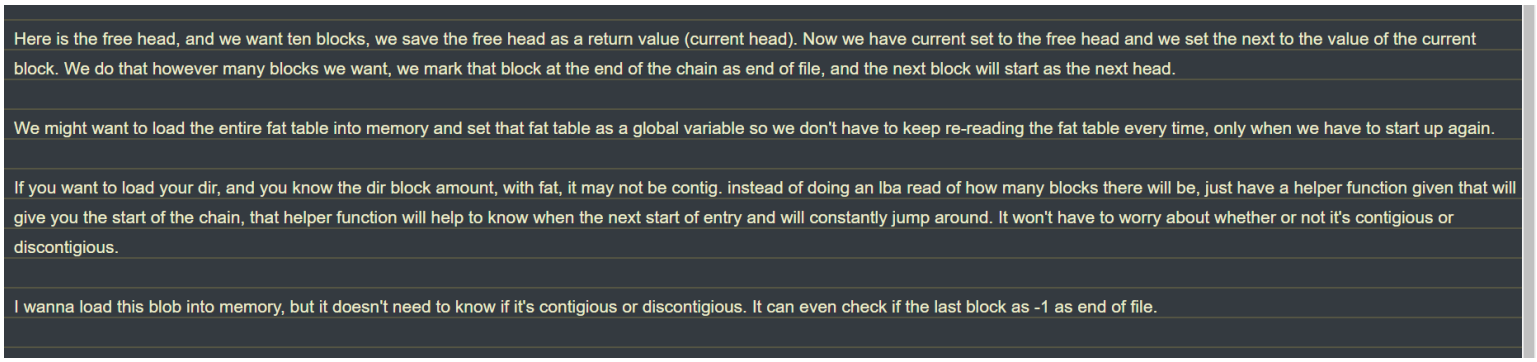
Issues and Resolutions:

- **Issue #1:** Our first issue as a group was having to start out with our given sections, more specifically with the free space management. What we came across was mainly confusion, since we did not have a solid plan to start out with. Some of our ideas came across, some of them were undecided, and it was a bit difficult to wrap our heads into translating certain parts into code.

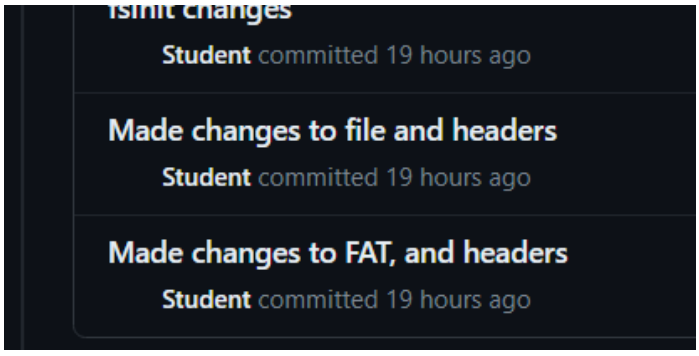
  Resolution: The main resolution that helped clear up this issue was by changing our own approach to how we would work together on starting the project. The biggest change that did help to clear up our confusion was by working in person together, rather than having to work in person. How this helped us was that by being able to physically show our own ideas on either a whiteboard, by talking directly to each other, or by being able to work directly hands on the code so we can see each other's own progress.
  - Another way we gained clarity on the assignment was also by attending office hours as a group, and keeping note of various pieces of feedback to what we should be doing, and how we should be thinking about our code. We would even keep those notes in the form of comments on a google doc, or in an online notepad.



- **Issue #2:** Another issue that was found in our group was certain commits not showing



user names and being listed as a Student.



Resolution #2: How this issue was fixed was by putting in the correct git credentials, mainly for Angelo. From there, the output for commits was corrected.

- **Issue #3:** One issue our group came across when we did our free space initializer. When making it, we decided to utilize a struct that contained two integers, one that shows which block is used and what is the next block for the free map. In addition, our free space map is utilizing an array, but when mallocing the amount of bytes it will take up based on the amount of blocks it will take up in our volume, we would get an initialize error

```
gcc -c -o fsshell.o fsshell.c -g -I.
gcc -c -o fsInit.o fsInit.c -g -I.
gcc -c -o FATInit.o FATInit.c -g -I.
FATInit.c: In function 'FATInitialize':
FATInit.c:52:31: error: invalid initializer
     blockIndicator FATMap[] = malloc(FATByteSize);
                               ^~~~~~
Makefile:58: recipe for target 'FATInit.o' failed
make: *** [FATInit.o] Error 1
```

Resolution #3: How we fixed this issue was when we realized that malloc returns a pointer value. If we used it, we should use a pointer value in order to malloc a certain size for our FATMap.

```
FATByteSize = blocksNeeded * BLOCK_SIZE; // For default, this is now 156672
blockIndicator *FATMap = malloc(FATByteSize);
```

HexDump Analysis:

- Volume Control Block (From virtual address 0x000200 to 0x0002F0):

```
dump.linux SampleVolume --start 1 --count 1
Dumping file SampleVolume, starting at block 1 for 1 block:

000200: 4B 4C 00 00 00 02 00 00  FB 4A 00 00 01 00 00 00 | KL......ûJ......
000210: 50 01 00 00 33 01 00 00  56 34 12 CE 0A 00 00 00 | P...3...V4.Î....
000220: 20 53 79 73 74 65 6D 20  50 61 72 74 69 74 69 6F |  System Partitio
000230: 6E 20 48 65 61 64 65 72  0A 0A 00 00 00 00 00 00 | n Header........
000240: 42 20 74 72 65 62 6F 52  00 96 98 00 00 00 00 00 | B treboR.ÿÿ.....
000250: 00 02 00 00 00 00 00 00  4B 4C 00 00 00 00 00 00 | ........KL......
000260: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000270: 52 6F 62 65 72 74 20 42  55 6E 74 69 74 6C 65 64 | Robert BUntitled
000280: 0A 0A 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 | ................
```

- The Hex dump for the Volume control block shows the values we initialized our vcb struct with in fsInit.c. The first four bytes(in pink) of the hex dump represent our first field in our vcb struct which is blocks for the number of blocks in the vcb. Since the hex dump is in little endian we read 4B 4C 00 00 as 00 00 4C 4B which is 19,531 which is the correct amount of blocks in our volume.
- The next 4 bytes in the hex dump(in blue) is blockSize 00000200 (we reverse because it is little endian) which is 512 when you translate it to decimal which is correct, that is how big our blockSize is.

- The next 4 bytes is the freeBlocks field in our vcb struct (in red) is 00004AFB which is 19195 blocks which represents the amount of free blocks we have left.
- The next 4 bytes is the fatStart (starting block) (in green) which translates to 00000001 -> 1 in decimal which is true since block 0 is the volume control block and the next block should be where our file allocation table starts.
- The next 4 bytes is the freeSpaceStart (in purple) which translates as 00000150 -> 336 so our freeSpace starts at block 336.
- The next 4 bytes is the rootDirStart (in brown) which translates from 00000133 -> 307, so our root dir starts at block 307 which is correct since our FAT takes up 306 blocks.
- The next 8 bytes is the signature since the signature field in our vcb (in yellow) is a long . When you convert 0000000ACE123456 to ascii you get our signature which is Robert B.

- FAT(Free Space Manager)(From Virtual Address 0x000400 - 0x0267F0):

```
000400: 00 00 00 00 00 00 00 00  02 00 00 00 01 00 00 00 | ................
000410: 03 00 00 00 01 00 00 00  04 00 00 00 01 00 00 00 | ................
000420: 05 00 00 00 01 00 00 00  06 00 00 00 01 00 00 00 | ................
000430: 07 00 00 00 01 00 00 00  08 00 00 00 01 00 00 00 | ................
000440: 09 00 00 00 01 00 00 00  0A 00 00 00 01 00 00 00 | ................
000450: 0B 00 00 00 01 00 00 00  0C 00 00 00 01 00 00 00 | ................
000460: 0D 00 00 00 01 00 00 00  0E 00 00 00 01 00 00 00 | ................
000470: 0F 00 00 00 01 00 00 00  10 00 00 00 01 00 00 00 | ................
000480: 11 00 00 00 01 00 00 00  12 00 00 00 01 00 00 00 | ................
000490: 13 00 00 00 01 00 00 00  14 00 00 00 01 00 00 00 | ................
0004A0: 15 00 00 00 01 00 00 00  16 00 00 00 01 00 00 00 | ................
0004B0: 17 00 00 00 01 00 00 00  18 00 00 00 01 00 00 00 | ................
0004C0: 19 00 00 00 01 00 00 00  1A 00 00 00 01 00 00 00 | ................
0004D0: 1B 00 00 00 01 00 00 00  1C 00 00 00 01 00 00 00 | ................
0004E0: 1D 00 00 00 01 00 00 00  1E 00 00 00 01 00 00 00 | ................
0004F0: 1F 00 00 00 01 00 00 00  20 00 00 00 01 00 00 00 | ........ .......

000500: 21 00 00 00 01 00 00 00  22 00 00 00 01 00 00 00 | !.......".......
000510: 23 00 00 00 01 00 00 00  24 00 00 00 01 00 00 00 | #.......$.......
000520: 25 00 00 00 01 00 00 00  26 00 00 00 01 00 00 00 | %.......&.......
000530: 27 00 00 00 01 00 00 00  28 00 00 00 01 00 00 00 | '.......(.......
000540: 29 00 00 00 01 00 00 00  2A 00 00 00 01 00 00 00 | ).......*.......
000550: 2B 00 00 00 01 00 00 00  2C 00 00 00 01 00 00 00 | +.......,.......
000560: 2D 00 00 00 01 00 00 00  2E 00 00 00 01 00 00 00 | -................
000570: 2F 00 00 00 01 00 00 00  30 00 00 00 01 00 00 00 | /.......0.......
000580: 31 00 00 00 01 00 00 00  32 00 00 00 01 00 00 00 | 1.......2.......
000590: 33 00 00 00 01 00 00 00  34 00 00 00 01 00 00 00 | 3.......4.......
0005A0: 35 00 00 00 01 00 00 00  36 00 00 00 01 00 00 00 | 5.......6.......
0005B0: 37 00 00 00 01 00 00 00  38 00 00 00 01 00 00 00 | 7.......8.......
0005C0: 39 00 00 00 01 00 00 00  3A 00 00 00 01 00 00 00 | 9.......:.......
0005D0: 3B 00 00 00 01 00 00 00  3C 00 00 00 01 00 00 00 | ;.......<.......
0005E0: 3D 00 00 00 01 00 00 00  3E 00 00 00 01 00 00 00 | =.......>.......
0005F0: 3F 00 00 00 01 00 00 00  40 00 00 00 01 00 00 00 | ?.......@.......
```

```
026600: 41 4C 00 00 00 00 00 00  42 4C 00 00 00 00 00 00 | AL......BL......
026610: 43 4C 00 00 00 00 00 00  44 4C 00 00 00 00 00 00 | CL......DL......
026620: 45 4C 00 00 00 00 00 00  46 4C 00 00 00 00 00 00 | EL......FL......
026630: 47 4C 00 00 00 00 00 00  48 4C 00 00 00 00 00 00 | GL......HL......
026640: 49 4C 00 00 00 00 00 00  4A 4C 00 00 00 00 00 00 | IL......JL......
026650: FE FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ????............
026660: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026670: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026680: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026690: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0266F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

026700: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026710: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026720: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026730: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026740: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026750: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026760: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026770: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026780: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
026790: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0267F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

- **Hex Address Interpreted:**
    - **(Virtual address: 0x026650)FEFFFFFF:** A part of the FAT Map that indicates where our entire volume ends at, because address 4A4C (when converted from little endian) converts 19530, which is block 19530 of our entire volume. And the rest of the addresses will indicate the previous blocks we are mapping to.

- Root Directory(From Virtual Address 0x026800 - 0x02A1F0):

```
026800: 2E 00 00 00 00 00 00 00  02 00 00 00 01 00 00 00 | ................
026810: 03 00 00 00 01 00 00 00  04 00 00 00 01 00 00 00 | ................
026820: 05 00 00 00 01 00 00 00  06 00 00 00 01 00 00 00 | ................
026830: 07 00 00 00 01 00 00 00  08 00 00 00 01 00 00 00 | ................
026840: 09 00 00 00 01 00 00 00  0A 00 00 00 01 00 00 00 | ................
026850: 0B 00 00 00 01 00 00 00  0C 00 00 00 01 00 00 00 | ................
026860: 0D 00 00 00 01 00 00 00  0E 00 00 00 01 00 00 00 | ................
026870: 0F 00 00 00 01 00 00 00  10 00 00 00 01 00 00 00 | ................
026880: 11 00 00 00 01 00 00 00  12 00 00 00 01 00 00 00 | ................
026890: 13 00 00 00 01 00 00 00  14 00 00 00 01 00 00 00 | ................
0268A0: 15 00 00 00 01 00 00 00  16 00 00 00 01 00 00 00 | ................
0268B0: 17 00 00 00 01 00 00 00  18 00 00 00 01 00 00 00 | ................
0268C0: 19 00 00 00 01 00 00 00  1A 00 00 00 01 00 00 00 | ................
0268D0: 1B 00 00 00 01 00 00 00  1C 00 00 00 01 00 00 00 | ................
0268E0: 1D 00 00 00 01 00 00 00  1E 00 00 00 01 00 00 00 | ................
0268F0: 1F 00 00 00 01 00 00 00  20 00 00 00 01 00 00 00 | ........ .......

026900: 33 01 00 00 D0 39 00 00  70 8C 3D 65 00 00 00 00 | 3...♦9..p♦=e....
026910: 70 8C 3D 65 00 00 00 00  70 8C 3D 65 00 00 00 00 | p♦=e....p♦=e....
026920: 01 00 00 00 01 00 00 00  2E 2E 00 00 01 00 00 00 | ................
026930: 27 00 00 00 01 00 00 00  28 00 00 00 01 00 00 00 | '.......(.......
026940: 29 00 00 00 01 00 00 00  2A 00 00 00 01 00 00 00 | ).......*.......
026950: 2B 00 00 00 01 00 00 00  2C 00 00 00 01 00 00 00 | +.......,.......
026960: 2D 00 00 00 01 00 00 00  2E 00 00 00 01 00 00 00 | -................
026970: 2F 00 00 00 01 00 00 00  30 00 00 00 01 00 00 00 | /.......0.......
026980: 31 00 00 00 01 00 00 00  32 00 00 00 01 00 00 00 | 1.......2.......
026990: 33 00 00 00 01 00 00 00  34 00 00 00 01 00 00 00 | 3.......4.......
0269A0: 35 00 00 00 01 00 00 00  36 00 00 00 01 00 00 00 | 5.......6.......
0269B0: 37 00 00 00 01 00 00 00  38 00 00 00 01 00 00 00 | 7.......8.......
0269C0: 39 00 00 00 01 00 00 00  3A 00 00 00 01 00 00 00 | 9.......:.......
0269D0: 3B 00 00 00 01 00 00 00  3C 00 00 00 01 00 00 00 | ;.......<.......
0269E0: 3D 00 00 00 01 00 00 00  3E 00 00 00 01 00 00 00 | =.......>.......
0269F0: 3F 00 00 00 01 00 00 00  40 00 00 00 01 00 00 00 | ?.......@.......
```

```
026A00: 41 00 00 00 01 00 00 00  42 00 00 00 01 00 00 00 | A.......B.......
026A10: 43 00 00 00 01 00 00 00  44 00 00 00 01 00 00 00 | C.......D.......
026A20: 45 00 00 00 01 00 00 00  33 01 00 00 D0 39 00 00 | E.......3...◆9..
026A30: 70 8C 3D 65 00 00 00 00  70 8C 3D 65 00 00 00 00 | p◆=e....p◆=e....
026A40: 70 8C 3D 65 00 00 00 00  01 00 00 00 01 00 00 00 | p◆=e............
026A50: 00 00 00 00 01 00 00 00  4C 00 00 00 01 00 00 00 | ........L.......
026A60: 4D 00 00 00 01 00 00 00  4E 00 00 00 01 00 00 00 | M.......N.......
026A70: 4F 00 00 00 01 00 00 00  50 00 00 00 01 00 00 00 | O.......P.......
026A80: 51 00 00 00 01 00 00 00  52 00 00 00 01 00 00 00 | Q.......R.......
026A90: 53 00 00 00 01 00 00 00  54 00 00 00 01 00 00 00 | S.......T.......
026AA0: 55 00 00 00 01 00 00 00  56 00 00 00 01 00 00 00 | U.......V.......
026AB0: 57 00 00 00 01 00 00 00  58 00 00 00 01 00 00 00 | W.......X.......
026AC0: 59 00 00 00 01 00 00 00  5A 00 00 00 01 00 00 00 | Y.......Z.......
026AD0: 5B 00 00 00 01 00 00 00  5C 00 00 00 01 00 00 00 | [.......\.......
026AE0: 5D 00 00 00 01 00 00 00  5E 00 00 00 01 00 00 00 | ].......^.......
026AF0: 5F 00 00 00 01 00 00 00  60 00 00 00 01 00 00 00 | _.......`.......

026B00: 61 00 00 00 01 00 00 00  62 00 00 00 01 00 00 00 | a.......b.......
026B10: 63 00 00 00 01 00 00 00  64 00 00 00 01 00 00 00 | c.......d.......
026B20: 65 00 00 00 01 00 00 00  66 00 00 00 01 00 00 00 | e.......f.......
026B30: 67 00 00 00 01 00 00 00  68 00 00 00 01 00 00 00 | g.......h.......
026B40: 69 00 00 00 01 00 00 00  6A 00 00 00 01 00 00 00 | i.......j.......
026B50: 6B 00 00 00 01 00 00 00  6C 00 00 00 01 00 00 00 | k.......l.......
026B60: 6D 00 00 00 01 00 00 00  6E 00 00 00 01 00 00 00 | m.......n.......
026B70: 6F 00 00 00 01 00 00 00  00 00 00 00 01 00 00 00 | o...............
026B80: 71 00 00 00 01 00 00 00  72 00 00 00 01 00 00 00 | q.......r.......
026B90: 73 00 00 00 01 00 00 00  74 00 00 00 01 00 00 00 | s.......t.......
026BA0: 75 00 00 00 01 00 00 00  76 00 00 00 01 00 00 00 | u.......v.......
026BB0: 77 00 00 00 01 00 00 00  78 00 00 00 01 00 00 00 | w.......x.......
026BC0: 79 00 00 00 01 00 00 00  7A 00 00 00 01 00 00 00 | y.......z.......
026BD0: 7B 00 00 00 01 00 00 00  7C 00 00 00 01 00 00 00 | {.......|.......
026BE0: 7D 00 00 00 01 00 00 00  7E 00 00 00 01 00 00 00 | }.......~.......
026BF0: 7F 00 00 00 01 00 00 00  80 00 00 00 01 00 00 00 | .......◆.......
```

```
026C00: 81 00 00 00 01 00 00 00  82 00 00 00 01 00 00 00 | ....????....????
026C10: 83 00 00 00 01 00 00 00  84 00 00 00 01 00 00 00 | ....????....????
026C20: 85 00 00 00 01 00 00 00  86 00 00 00 01 00 00 00 | ....????....????
026C30: 87 00 00 00 01 00 00 00  88 00 00 00 01 00 00 00 | ....????....????
026C40: 89 00 00 00 01 00 00 00  8A 00 00 00 01 00 00 00 | ....????....????
026C50: 8B 00 00 00 01 00 00 00  8C 00 00 00 01 00 00 00 | ....????....????
026C60: 8D 00 00 00 01 00 00 00  8E 00 00 00 01 00 00 00 | ....????....????
026C70: 8F 00 00 00 01 00 00 00  90 00 00 00 01 00 00 00 | ....????....????
026C80: 91 00 00 00 01 00 00 00  92 00 00 00 01 00 00 00 | ....????....????
026C90: 93 00 00 00 01 00 00 00  94 00 00 00 01 00 00 00 | ....????....????
026CA0: 00 00 00 00 01 00 00 00  96 00 00 00 01 00 00 00 | ........????....
026CB0: 97 00 00 00 01 00 00 00  98 00 00 00 01 00 00 00 | ....????....????
026CC0: 99 00 00 00 01 00 00 00  9A 00 00 00 01 00 00 00 | ....????....????
026CD0: 9B 00 00 00 01 00 00 00  9C 00 00 00 01 00 00 00 | ....????....????
026CE0: 9D 00 00 00 01 00 00 00  9E 00 00 00 01 00 00 00 | ....????....????
026CF0: 9F 00 00 00 01 00 00 00  A0 00 00 00 01 00 00 00 | ....????....????

026D00: A1 00 00 00 01 00 00 00  A2 00 00 00 01 00 00 00 | ....????....????
026D10: A3 00 00 00 01 00 00 00  A4 00 00 00 01 00 00 00 | ....????....????
026D20: A5 00 00 00 01 00 00 00  A6 00 00 00 01 00 00 00 | ....????....????
026D30: A7 00 00 00 01 00 00 00  A8 00 00 00 01 00 00 00 | ....????....????
026D40: A9 00 00 00 01 00 00 00  AA 00 00 00 01 00 00 00 | ....????....????
026D50: AB 00 00 00 01 00 00 00  AC 00 00 00 01 00 00 00 | ....????....????
026D60: AD 00 00 00 01 00 00 00  AE 00 00 00 01 00 00 00 | ....????....????
026D70: AF 00 00 00 01 00 00 00  B0 00 00 00 01 00 00 00 | ....????....????
026D80: B1 00 00 00 01 00 00 00  B2 00 00 00 01 00 00 00 | ....????....????
026D90: B3 00 00 00 01 00 00 00  B4 00 00 00 01 00 00 00 | ....????....????
026DA0: B5 00 00 00 01 00 00 00  B6 00 00 00 01 00 00 00 | ....????....????
026DB0: B7 00 00 00 01 00 00 00  B8 00 00 00 01 00 00 00 | ....????....????
026DC0: B9 00 00 00 01 00 00 00  00 00 00 00 01 00 00 00 | ....????........
026DD0: BB 00 00 00 01 00 00 00  BC 00 00 00 01 00 00 00 | ....????....????
026DE0: BD 00 00 00 01 00 00 00  BE 00 00 00 01 00 00 00 | ....????....????
026DF0: BF 00 00 00 01 00 00 00  C0 00 00 00 01 00 00 00 | ....????....????
```

```
026E00: C1 00 00 00 01 00 00 00  C2 00 00 00 01 00 00 00 | ♦.......♦.......
026E10: C3 00 00 00 01 00 00 00  C4 00 00 00 01 00 00 00 | ♦.......♦.......
026E20: C5 00 00 00 01 00 00 00  C6 00 00 00 01 00 00 00 | ♦.......♦.......
026E30: C7 00 00 00 01 00 00 00  C8 00 00 00 01 00 00 00 | ♦.......♦.......
026E40: C9 00 00 00 01 00 00 00  CA 00 00 00 01 00 00 00 | ♦.......♦.......
026E50: CB 00 00 00 01 00 00 00  CC 00 00 00 01 00 00 00 | ♦.......♦.......
026E60: CD 00 00 00 01 00 00 00  CE 00 00 00 01 00 00 00 | ♦.......♦.......
026E70: CF 00 00 00 01 00 00 00  D0 00 00 00 01 00 00 00 | ♦.......♦.......
026E80: D1 00 00 00 01 00 00 00  D2 00 00 00 01 00 00 00 | ♦.......♦.......
026E90: D3 00 00 00 01 00 00 00  D4 00 00 00 01 00 00 00 | ♦.......♦.......
026EA0: D5 00 00 00 01 00 00 00  D6 00 00 00 01 00 00 00 | ♦.......♦.......
026EB0: D7 00 00 00 01 00 00 00  D8 00 00 00 01 00 00 00 | ♦.......♦.......
026EC0: D9 00 00 00 01 00 00 00  DA 00 00 00 01 00 00 00 | ♦.......♦.......
026ED0: DB 00 00 00 01 00 00 00  DC 00 00 00 01 00 00 00 | ♦.......♦.......
026EE0: DD 00 00 00 01 00 00 00  DE 00 00 00 01 00 00 00 | ♦.......♦.......
026EF0: 00 00 00 00 01 00 00 00  E0 00 00 00 01 00 00 00 | ........♦.......

026F00: E1 00 00 00 01 00 00 00  E2 00 00 00 01 00 00 00 | ♦.......♦.......
026F10: E3 00 00 00 01 00 00 00  E4 00 00 00 01 00 00 00 | ♦.......♦.......
026F20: E5 00 00 00 01 00 00 00  E6 00 00 00 01 00 00 00 | ♦.......♦.......
026F30: E7 00 00 00 01 00 00 00  E8 00 00 00 01 00 00 00 | ♦.......♦.......
026F40: E9 00 00 00 01 00 00 00  EA 00 00 00 01 00 00 00 | ♦.......♦.......
026F50: EB 00 00 00 01 00 00 00  EC 00 00 00 01 00 00 00 | ♦.......♦.......
026F60: ED 00 00 00 01 00 00 00  EE 00 00 00 01 00 00 00 | ♦.......♦.......
026F70: EF 00 00 00 01 00 00 00  F0 00 00 00 01 00 00 00 | ♦.......♦.......
026F80: F1 00 00 00 01 00 00 00  F2 00 00 00 01 00 00 00 | ♦.......♦.......
026F90: F3 00 00 00 01 00 00 00  F4 00 00 00 01 00 00 00 | ♦.......♦.......
026FA0: F5 00 00 00 01 00 00 00  F6 00 00 00 01 00 00 00 | ♦.......♦.......
026FB0: F7 00 00 00 01 00 00 00  F8 00 00 00 01 00 00 00 | ♦.......♦.......
026FC0: F9 00 00 00 01 00 00 00  FA 00 00 00 01 00 00 00 | ♦.......♦.......
026FD0: FB 00 00 00 01 00 00 00  FC 00 00 00 01 00 00 00 | ♦.......♦.......
026FE0: FD 00 00 00 01 00 00 00  FE 00 00 00 01 00 00 00 | ♦.......♦.......
026FF0: FF 00 00 00 01 00 00 00  00 01 00 00 01 00 00 00 | ♦...............
```

```
027000: 01 01 00 00 01 00 00 00  02 01 00 00 01 00 00 00 | ................
027010: 03 01 00 00 01 00 00 00  00 01 00 00 01 00 00 00 | ................
027020: 05 01 00 00 01 00 00 00  06 01 00 00 01 00 00 00 | ................
027030: 07 01 00 00 01 00 00 00  08 01 00 00 01 00 00 00 | ................
027040: 09 01 00 00 01 00 00 00  0A 01 00 00 01 00 00 00 | ................
027050: 0B 01 00 00 01 00 00 00  0C 01 00 00 01 00 00 00 | ................
027060: 0D 01 00 00 01 00 00 00  0E 01 00 00 01 00 00 00 | ................
027070: 0F 01 00 00 01 00 00 00  10 01 00 00 01 00 00 00 | ................
027080: 11 01 00 00 01 00 00 00  12 01 00 00 01 00 00 00 | ................
027090: 13 01 00 00 01 00 00 00  14 01 00 00 01 00 00 00 | ................
0270A0: 15 01 00 00 01 00 00 00  16 01 00 00 01 00 00 00 | ................
0270B0: 17 01 00 00 01 00 00 00  18 01 00 00 01 00 00 00 | ................
0270C0: 19 01 00 00 01 00 00 00  1A 01 00 00 01 00 00 00 | ................
0270D0: 1B 01 00 00 01 00 00 00  1C 01 00 00 01 00 00 00 | ................
0270E0: 1D 01 00 00 01 00 00 00  1E 01 00 00 01 00 00 00 | ................
0270F0: 1F 01 00 00 01 00 00 00  20 01 00 00 01 00 00 00 | ........ .......

027100: 21 01 00 00 01 00 00 00  22 01 00 00 01 00 00 00 | !.......".......
027110: 23 01 00 00 01 00 00 00  24 01 00 00 01 00 00 00 | #.......$.......
027120: 25 01 00 00 01 00 00 00  26 01 00 00 01 00 00 00 | %.......&.......
027130: 27 01 00 00 01 00 00 00  28 01 00 00 01 00 00 00 | '.......(.......
027140: 00 01 00 00 01 00 00 00  2A 01 00 00 01 00 00 00 | ........*.......
027150: 2B 01 00 00 01 00 00 00  2C 01 00 00 01 00 00 00 | +.......,.......
027160: 2D 01 00 00 01 00 00 00  2E 01 00 00 01 00 00 00 | -...............
027170: 2F 01 00 00 01 00 00 00  30 01 00 00 01 00 00 00 | /.......0.......
027180: 31 01 00 00 01 00 00 00  32 01 00 00 01 00 00 00 | 1.......2.......
027190: 33 01 00 00 01 00 00 00  34 01 00 00 00 00 00 00 | 3.......4.......
0271A0: 35 01 00 00 00 00 00 00  36 01 00 00 00 00 00 00 | 5.......6.......
0271B0: 37 01 00 00 00 00 00 00  38 01 00 00 00 00 00 00 | 7.......8.......
0271C0: 39 01 00 00 00 00 00 00  3A 01 00 00 00 00 00 00 | 9.......:.......
0271D0: 3B 01 00 00 00 00 00 00  3C 01 00 00 00 00 00 00 | ;.......<.......
0271E0: 3D 01 00 00 00 00 00 00  3E 01 00 00 00 00 00 00 | =.......>.......
0271F0: 3F 01 00 00 00 00 00 00  40 01 00 00 00 00 00 00 | ?.......@.......
```

```
027200: 41 01 00 00 00 00 00 00  42 01 00 00 00 00 00 00 | A.......B.......
027210: 43 01 00 00 00 00 00 00  44 01 00 00 00 00 00 00 | C.......D.......
027220: 45 01 00 00 00 00 00 00  46 01 00 00 00 00 00 00 | E.......F.......
027230: 47 01 00 00 00 00 00 00  48 01 00 00 00 00 00 00 | G.......H.......
027240: 49 01 00 00 00 00 00 00  4A 01 00 00 00 00 00 00 | I.......J.......
027250: 4B 01 00 00 00 00 00 00  4C 01 00 00 00 00 00 00 | K.......L.......
027260: 4D 01 00 00 00 00 00 00  00 01 00 00 00 00 00 00 | M...............
027270: 4F 01 00 00 00 00 00 00  50 01 00 00 00 00 00 00 | O.......P.......
027280: 51 01 00 00 00 00 00 00  52 01 00 00 00 00 00 00 | Q.......R.......
027290: 53 01 00 00 00 00 00 00  54 01 00 00 00 00 00 00 | S.......T.......
0272A0: 55 01 00 00 00 00 00 00  56 01 00 00 00 00 00 00 | U.......V.......
0272B0: 57 01 00 00 00 00 00 00  58 01 00 00 00 00 00 00 | W.......X.......
0272C0: 59 01 00 00 00 00 00 00  5A 01 00 00 00 00 00 00 | Y.......Z.......
0272D0: 5B 01 00 00 00 00 00 00  5C 01 00 00 00 00 00 00 | [.......\.......
0272E0: 5D 01 00 00 00 00 00 00  5E 01 00 00 00 00 00 00 | ].......^.......
0272F0: 5F 01 00 00 00 00 00 00  60 01 00 00 00 00 00 00 | _.......`.......

027300: 61 01 00 00 00 00 00 00  62 01 00 00 00 00 00 00 | a.......b.......
027310: 63 01 00 00 00 00 00 00  64 01 00 00 00 00 00 00 | c.......d.......
027320: 65 01 00 00 00 00 00 00  66 01 00 00 00 00 00 00 | e.......f.......
027330: 67 01 00 00 00 00 00 00  68 01 00 00 00 00 00 00 | g.......h.......
027340: 69 01 00 00 00 00 00 00  6A 01 00 00 00 00 00 00 | i.......j.......
027350: 6B 01 00 00 00 00 00 00  6C 01 00 00 00 00 00 00 | k.......l.......
027360: 6D 01 00 00 00 00 00 00  6E 01 00 00 00 00 00 00 | m.......n.......
027370: 6F 01 00 00 00 00 00 00  70 01 00 00 00 00 00 00 | o.......p.......
027380: 71 01 00 00 00 00 00 00  72 01 00 00 00 00 00 00 | q.......r.......
027390: 00 01 00 00 00 00 00 00  74 01 00 00 00 00 00 00 | ........t.......
0273A0: 75 01 00 00 00 00 00 00  76 01 00 00 00 00 00 00 | u.......v.......
0273B0: 77 01 00 00 00 00 00 00  78 01 00 00 00 00 00 00 | w.......x.......
0273C0: 79 01 00 00 00 00 00 00  7A 01 00 00 00 00 00 00 | y.......z.......
0273D0: 7B 01 00 00 00 00 00 00  7C 01 00 00 00 00 00 00 | {.......|.......
0273E0: 7D 01 00 00 00 00 00 00  7E 01 00 00 00 00 00 00 | }.......~.......
0273F0: 7F 01 00 00 00 00 00 00  80 01 00 00 00 00 00 00 | ................
```

```
027400: 81 01 00 00 00 00 00 00  82 01 00 00 00 00 00 00 | ◆.......◆.......
027410: 83 01 00 00 00 00 00 00  84 01 00 00 00 00 00 00 | ◆.......◆.......
027420: 85 01 00 00 00 00 00 00  86 01 00 00 00 00 00 00 | ◆.......◆.......
027430: 87 01 00 00 00 00 00 00  88 01 00 00 00 00 00 00 | ◆.......◆.......
027440: 89 01 00 00 00 00 00 00  8A 01 00 00 00 00 00 00 | ◆.......◆.......
027450: 8B 01 00 00 00 00 00 00  8C 01 00 00 00 00 00 00 | ◆.......◆.......
027460: 8D 01 00 00 00 00 00 00  8E 01 00 00 00 00 00 00 | ◆.......◆.......
027470: 8F 01 00 00 00 00 00 00  90 01 00 00 00 00 00 00 | ◆.......◆.......
027480: 91 01 00 00 00 00 00 00  92 01 00 00 00 00 00 00 | ◆.......◆.......
027490: 93 01 00 00 00 00 00 00  94 01 00 00 00 00 00 00 | ◆.......◆.......
0274A0: 95 01 00 00 00 00 00 00  96 01 00 00 00 00 00 00 | ◆.......◆.......
0274B0: 97 01 00 00 00 00 00 00  00 01 00 00 00 00 00 00 | ◆...............
0274C0: 99 01 00 00 00 00 00 00  9A 01 00 00 00 00 00 00 | ◆.......◆.......
0274D0: 9B 01 00 00 00 00 00 00  9C 01 00 00 00 00 00 00 | ◆.......◆.......
0274E0: 9D 01 00 00 00 00 00 00  9E 01 00 00 00 00 00 00 | ◆.......◆.......
0274F0: 9F 01 00 00 00 00 00 00  A0 01 00 00 00 00 00 00 | ◆.......◆.......

027500: A1 01 00 00 00 00 00 00  A2 01 00 00 00 00 00 00 | ◆.......◆.......
027510: A3 01 00 00 00 00 00 00  A4 01 00 00 00 00 00 00 | ◆.......◆.......
027520: A5 01 00 00 00 00 00 00  A6 01 00 00 00 00 00 00 | ◆.......◆.......
027530: A7 01 00 00 00 00 00 00  A8 01 00 00 00 00 00 00 | ◆.......◆.......
027540: A9 01 00 00 00 00 00 00  AA 01 00 00 00 00 00 00 | ◆.......◆.......
027550: AB 01 00 00 00 00 00 00  AC 01 00 00 00 00 00 00 | ◆.......◆.......
027560: AD 01 00 00 00 00 00 00  AE 01 00 00 00 00 00 00 | ◆.......◆.......
027570: AF 01 00 00 00 00 00 00  B0 01 00 00 00 00 00 00 | ◆.......◆.......
027580: B1 01 00 00 00 00 00 00  B2 01 00 00 00 00 00 00 | ◆.......◆.......
027590: B3 01 00 00 00 00 00 00  B4 01 00 00 00 00 00 00 | ◆.......◆.......
0275A0: B5 01 00 00 00 00 00 00  B6 01 00 00 00 00 00 00 | ◆.......◆.......
0275B0: B7 01 00 00 00 00 00 00  B8 01 00 00 00 00 00 00 | ◆.......◆.......
0275C0: B9 01 00 00 00 00 00 00  BA 01 00 00 00 00 00 00 | ◆.......◆.......
0275D0: BB 01 00 00 00 00 00 00  BC 01 00 00 00 00 00 00 | ◆.......◆.......
0275E0: 00 01 00 00 00 00 00 00  BE 01 00 00 00 00 00 00 |  ........◆.......
0275F0: BF 01 00 00 00 00 00 00  C0 01 00 00 00 00 00 00 | ◆.......◆.......
```

```
027600: C1 01 00 00 00 00 00 00  C2 01 00 00 00 00 00 00 | ....... .......
027610: C3 01 00 00 00 00 00 00  C4 01 00 00 00 00 00 00 | ....... .......
027620: C5 01 00 00 00 00 00 00  C6 01 00 00 00 00 00 00 | ....... .......
027630: C7 01 00 00 00 00 00 00  C8 01 00 00 00 00 00 00 | ....... .......
027640: C9 01 00 00 00 00 00 00  CA 01 00 00 00 00 00 00 | ....... .......
027650: CB 01 00 00 00 00 00 00  CC 01 00 00 00 00 00 00 | ....... .......
027660: CD 01 00 00 00 00 00 00  CE 01 00 00 00 00 00 00 | ....... .......
027670: CF 01 00 00 00 00 00 00  D0 01 00 00 00 00 00 00 | ....... .......
027680: D1 01 00 00 00 00 00 00  D2 01 00 00 00 00 00 00 | ....... .......
027690: D3 01 00 00 00 00 00 00  D4 01 00 00 00 00 00 00 | ....... .......
0276A0: D5 01 00 00 00 00 00 00  D6 01 00 00 00 00 00 00 | ....... .......
0276B0: D7 01 00 00 00 00 00 00  D8 01 00 00 00 00 00 00 | ....... .......
0276C0: D9 01 00 00 00 00 00 00  DA 01 00 00 00 00 00 00 | ....... .......
0276D0: DB 01 00 00 00 00 00 00  DC 01 00 00 00 00 00 00 | ....... .......
0276E0: DD 01 00 00 00 00 00 00  DE 01 00 00 00 00 00 00 | ....... .......
0276F0: DF 01 00 00 00 00 00 00  E0 01 00 00 00 00 00 00 | ....... .......

027700: E1 01 00 00 00 00 00 00  00 01 00 00 00 00 00 00 | ....... .......
027710: E3 01 00 00 00 00 00 00  E4 01 00 00 00 00 00 00 | ....... .......
027720: E5 01 00 00 00 00 00 00  E6 01 00 00 00 00 00 00 | ....... .......
027730: E7 01 00 00 00 00 00 00  E8 01 00 00 00 00 00 00 | ....... .......
027740: E9 01 00 00 00 00 00 00  EA 01 00 00 00 00 00 00 | ....... .......
027750: EB 01 00 00 00 00 00 00  EC 01 00 00 00 00 00 00 | ....... .......
027760: ED 01 00 00 00 00 00 00  EE 01 00 00 00 00 00 00 | ....... .......
027770: EF 01 00 00 00 00 00 00  F0 01 00 00 00 00 00 00 | ....... .......
027780: F1 01 00 00 00 00 00 00  F2 01 00 00 00 00 00 00 | ....... .......
027790: F3 01 00 00 00 00 00 00  F4 01 00 00 00 00 00 00 | ....... .......
0277A0: F5 01 00 00 00 00 00 00  F6 01 00 00 00 00 00 00 | ....... .......
0277B0: F7 01 00 00 00 00 00 00  F8 01 00 00 00 00 00 00 | ....... .......
0277C0: F9 01 00 00 00 00 00 00  FA 01 00 00 00 00 00 00 | ....... .......
0277D0: FB 01 00 00 00 00 00 00  FC 01 00 00 00 00 00 00 | ....... .......
0277E0: FD 01 00 00 00 00 00 00  FE 01 00 00 00 00 00 00 | ....... .......
0277F0: FF 01 00 00 00 00 00 00  00 02 00 00 00 00 00 00 | ....... .......
```

```
027800: 01 02 00 00 00 00 00 00  02 02 00 00 00 00 00 00 | ................
027810: 03 02 00 00 00 00 00 00  04 02 00 00 00 00 00 00 | ................
027820: 05 02 00 00 00 00 00 00  06 02 00 00 00 00 00 00 | ................
027830: 00 02 00 00 00 00 00 00  08 02 00 00 00 00 00 00 | ................
027840: 09 02 00 00 00 00 00 00  0A 02 00 00 00 00 00 00 | ................
027850: 0B 02 00 00 00 00 00 00  0C 02 00 00 00 00 00 00 | ................
027860: 0D 02 00 00 00 00 00 00  0E 02 00 00 00 00 00 00 | ................
027870: 0F 02 00 00 00 00 00 00  10 02 00 00 00 00 00 00 | ................
027880: 11 02 00 00 00 00 00 00  12 02 00 00 00 00 00 00 | ................
027890: 13 02 00 00 00 00 00 00  14 02 00 00 00 00 00 00 | ................
0278A0: 15 02 00 00 00 00 00 00  16 02 00 00 00 00 00 00 | ................
0278B0: 17 02 00 00 00 00 00 00  18 02 00 00 00 00 00 00 | ................
0278C0: 19 02 00 00 00 00 00 00  1A 02 00 00 00 00 00 00 | ................
0278D0: 1B 02 00 00 00 00 00 00  1C 02 00 00 00 00 00 00 | ................
0278E0: 1D 02 00 00 00 00 00 00  1E 02 00 00 00 00 00 00 | ................
0278F0: 1F 02 00 00 00 00 00 00  20 02 00 00 00 00 00 00 | ........ .......

027900: 21 02 00 00 00 00 00 00  22 02 00 00 00 00 00 00 | !.......".......
027910: 23 02 00 00 00 00 00 00  24 02 00 00 00 00 00 00 | #.......$.......
027920: 25 02 00 00 00 00 00 00  26 02 00 00 00 00 00 00 | %.......&.......
027930: 27 02 00 00 00 00 00 00  28 02 00 00 00 00 00 00 | '.......(.......
027940: 29 02 00 00 00 00 00 00  2A 02 00 00 00 00 00 00 | ).......*.......
027950: 2B 02 00 00 00 00 00 00  00 02 00 00 00 00 00 00 | +................
027960: 2D 02 00 00 00 00 00 00  2E 02 00 00 00 00 00 00 | -................
027970: 2F 02 00 00 00 00 00 00  30 02 00 00 00 00 00 00 | /.......0.......
027980: 31 02 00 00 00 00 00 00  32 02 00 00 00 00 00 00 | 1.......2.......
027990: 33 02 00 00 00 00 00 00  34 02 00 00 00 00 00 00 | 3.......4.......
0279A0: 35 02 00 00 00 00 00 00  36 02 00 00 00 00 00 00 | 5.......6.......
0279B0: 37 02 00 00 00 00 00 00  38 02 00 00 00 00 00 00 | 7.......8.......
0279C0: 39 02 00 00 00 00 00 00  3A 02 00 00 00 00 00 00 | 9.......:.......
0279D0: 3B 02 00 00 00 00 00 00  3C 02 00 00 00 00 00 00 | ;.......<.......
0279E0: 3D 02 00 00 00 00 00 00  3E 02 00 00 00 00 00 00 | =.......>.......
0279F0: 3F 02 00 00 00 00 00 00  40 02 00 00 00 00 00 00 | ?.......@.......
```

```
027A00: 41 02 00 00 00 00 00 00  42 02 00 00 00 00 00 00 | A.......B.......
027A10: 43 02 00 00 00 00 00 00  44 02 00 00 00 00 00 00 | C.......D.......
027A20: 45 02 00 00 00 00 00 00  46 02 00 00 00 00 00 00 | E.......F.......
027A30: 47 02 00 00 00 00 00 00  48 02 00 00 00 00 00 00 | G.......H.......
027A40: 49 02 00 00 00 00 00 00  4A 02 00 00 00 00 00 00 | I.......J.......
027A50: 4B 02 00 00 00 00 00 00  4C 02 00 00 00 00 00 00 | K.......L.......
027A60: 4D 02 00 00 00 00 00 00  4E 02 00 00 00 00 00 00 | M.......N.......
027A70: 4F 02 00 00 00 00 00 00  50 02 00 00 00 00 00 00 | O.......P.......
027A80: 00 02 00 00 00 00 00 00  52 02 00 00 00 00 00 00 | ........R.......
027A90: 53 02 00 00 00 00 00 00  54 02 00 00 00 00 00 00 | S.......T.......
027AA0: 55 02 00 00 00 00 00 00  56 02 00 00 00 00 00 00 | U.......V.......
027AB0: 57 02 00 00 00 00 00 00  58 02 00 00 00 00 00 00 | W.......X.......
027AC0: 59 02 00 00 00 00 00 00  5A 02 00 00 00 00 00 00 | Y.......Z.......
027AD0: 5B 02 00 00 00 00 00 00  5C 02 00 00 00 00 00 00 | [.......\.......
027AE0: 5D 02 00 00 00 00 00 00  5E 02 00 00 00 00 00 00 | ].......^.......
027AF0: 5F 02 00 00 00 00 00 00  60 02 00 00 00 00 00 00 | _.......`.......

027B00: 61 02 00 00 00 00 00 00  62 02 00 00 00 00 00 00 | a.......b.......
027B10: 63 02 00 00 00 00 00 00  64 02 00 00 00 00 00 00 | c.......d.......
027B20: 65 02 00 00 00 00 00 00  66 02 00 00 00 00 00 00 | e.......f.......
027B30: 67 02 00 00 00 00 00 00  68 02 00 00 00 00 00 00 | g.......h.......
027B40: 69 02 00 00 00 00 00 00  6A 02 00 00 00 00 00 00 | i.......j.......
027B50: 6B 02 00 00 00 00 00 00  6C 02 00 00 00 00 00 00 | k.......l.......
027B60: 6D 02 00 00 00 00 00 00  6E 02 00 00 00 00 00 00 | m.......n.......
027B70: 6F 02 00 00 00 00 00 00  70 02 00 00 00 00 00 00 | o.......p.......
027B80: 71 02 00 00 00 00 00 00  72 02 00 00 00 00 00 00 | q.......r.......
027B90: 73 02 00 00 00 00 00 00  74 02 00 00 00 00 00 00 | s.......t.......
027BA0: 75 02 00 00 00 00 00 00  00 02 00 00 00 00 00 00 | u...............
027BB0: 77 02 00 00 00 00 00 00  78 02 00 00 00 00 00 00 | w.......x.......
027BC0: 79 02 00 00 00 00 00 00  7A 02 00 00 00 00 00 00 | y.......z.......
027BD0: 7B 02 00 00 00 00 00 00  7C 02 00 00 00 00 00 00 | {.......|.......
027BE0: 7D 02 00 00 00 00 00 00  7E 02 00 00 00 00 00 00 | }.......~.......
027BF0: 7F 02 00 00 00 00 00 00  80 02 00 00 00 00 00 00 | .......?.......
```

```
027C00: 81 02 00 00 00 00 00 00  82 02 00 00 00 00 00 00 | ◆.......◆.......
027C10: 83 02 00 00 00 00 00 00  84 02 00 00 00 00 00 00 | ◆.......◆.......
027C20: 85 02 00 00 00 00 00 00  86 02 00 00 00 00 00 00 | ◆.......◆.......
027C30: 87 02 00 00 00 00 00 00  88 02 00 00 00 00 00 00 | ◆.......◆.......
027C40: 89 02 00 00 00 00 00 00  8A 02 00 00 00 00 00 00 | ◆.......◆.......
027C50: 8B 02 00 00 00 00 00 00  8C 02 00 00 00 00 00 00 | ◆.......◆.......
027C60: 8D 02 00 00 00 00 00 00  8E 02 00 00 00 00 00 00 | ◆.......◆.......
027C70: 8F 02 00 00 00 00 00 00  90 02 00 00 00 00 00 00 | ◆.......◆.......
027C80: 91 02 00 00 00 00 00 00  92 02 00 00 00 00 00 00 | ◆.......◆.......
027C90: 93 02 00 00 00 00 00 00  94 02 00 00 00 00 00 00 | ◆.......◆.......
027CA0: 95 02 00 00 00 00 00 00  96 02 00 00 00 00 00 00 | ◆.......◆.......
027CB0: 97 02 00 00 00 00 00 00  98 02 00 00 00 00 00 00 | ◆.......◆.......
027CC0: 99 02 00 00 00 00 00 00  9A 02 00 00 00 00 00 00 | ◆.......◆.......
027CD0: 00 02 00 00 00 00 00 00  9C 02 00 00 00 00 00 00 |  .......◆.......
027CE0: 9D 02 00 00 00 00 00 00  9E 02 00 00 00 00 00 00 | ◆.......◆.......
027CF0: 9F 02 00 00 00 00 00 00  A0 02 00 00 00 00 00 00 | ◆.......◆.......

027D00: A1 02 00 00 00 00 00 00  A2 02 00 00 00 00 00 00 | ◆.......◆.......
027D10: A3 02 00 00 00 00 00 00  A4 02 00 00 00 00 00 00 | ◆.......◆.......
027D20: A5 02 00 00 00 00 00 00  A6 02 00 00 00 00 00 00 | ◆.......◆.......
027D30: A7 02 00 00 00 00 00 00  A8 02 00 00 00 00 00 00 | ◆.......◆.......
027D40: A9 02 00 00 00 00 00 00  AA 02 00 00 00 00 00 00 | ◆.......◆.......
027D50: AB 02 00 00 00 00 00 00  AC 02 00 00 00 00 00 00 | ◆.......◆.......
027D60: AD 02 00 00 00 00 00 00  AE 02 00 00 00 00 00 00 | ◆.......◆.......
027D70: AF 02 00 00 00 00 00 00  B0 02 00 00 00 00 00 00 | ◆.......◆.......
027D80: B1 02 00 00 00 00 00 00  B2 02 00 00 00 00 00 00 | ◆.......◆.......
027D90: B3 02 00 00 00 00 00 00  B4 02 00 00 00 00 00 00 | ◆.......◆.......
027DA0: B5 02 00 00 00 00 00 00  B6 02 00 00 00 00 00 00 | ◆.......◆.......
027DB0: B7 02 00 00 00 00 00 00  B8 02 00 00 00 00 00 00 | ◆.......◆.......
027DC0: B9 02 00 00 00 00 00 00  BA 02 00 00 00 00 00 00 | ◆.......◆.......
027DD0: BB 02 00 00 00 00 00 00  BC 02 00 00 00 00 00 00 | ◆.......◆.......
027DE0: BD 02 00 00 00 00 00 00  BE 02 00 00 00 00 00 00 | ◆.......◆.......
027DF0: BF 02 00 00 00 00 00 00  00 02 00 00 00 00 00 00 | ◆...............
```

```
027E00: C1 02 00 00 00 00 00 00  C2 02 00 00 00 00 00 00 | ?.......?.......
027E10: C3 02 00 00 00 00 00 00  C4 02 00 00 00 00 00 00 | ?.......?.......
027E20: C5 02 00 00 00 00 00 00  C6 02 00 00 00 00 00 00 | ?.......?.......
027E30: C7 02 00 00 00 00 00 00  C8 02 00 00 00 00 00 00 | ?.......?.......
027E40: C9 02 00 00 00 00 00 00  CA 02 00 00 00 00 00 00 | ?.......?.......
027E50: CB 02 00 00 00 00 00 00  CC 02 00 00 00 00 00 00 | ?.......?.......
027E60: CD 02 00 00 00 00 00 00  CE 02 00 00 00 00 00 00 | ?.......?.......
027E70: CF 02 00 00 00 00 00 00  D0 02 00 00 00 00 00 00 | ?.......?.......
027E80: D1 02 00 00 00 00 00 00  D2 02 00 00 00 00 00 00 | ?.......?.......
027E90: D3 02 00 00 00 00 00 00  D4 02 00 00 00 00 00 00 | ?.......?.......
027EA0: D5 02 00 00 00 00 00 00  D6 02 00 00 00 00 00 00 | ?.......?.......
027EB0: D7 02 00 00 00 00 00 00  D8 02 00 00 00 00 00 00 | ?.......?.......
027EC0: D9 02 00 00 00 00 00 00  DA 02 00 00 00 00 00 00 | ?.......?.......
027ED0: DB 02 00 00 00 00 00 00  DC 02 00 00 00 00 00 00 | ?.......?.......
027EE0: DD 02 00 00 00 00 00 00  DE 02 00 00 00 00 00 00 | ?.......?.......
027EF0: DF 02 00 00 00 00 00 00  E0 02 00 00 00 00 00 00 | ?.......?.......

027F00: E1 02 00 00 00 00 00 00  E2 02 00 00 00 00 00 00 | ?.......?.......
027F10: E3 02 00 00 00 00 00 00  E4 02 00 00 00 00 00 00 | ?.......?.......
027F20: 00 02 00 00 00 00 00 00  E6 02 00 00 00 00 00 00 |  .......?.......
027F30: E7 02 00 00 00 00 00 00  E8 02 00 00 00 00 00 00 | ?.......?.......
027F40: E9 02 00 00 00 00 00 00  EA 02 00 00 00 00 00 00 | ?.......?.......
027F50: EB 02 00 00 00 00 00 00  EC 02 00 00 00 00 00 00 | ?.......?.......
027F60: ED 02 00 00 00 00 00 00  EE 02 00 00 00 00 00 00 | ?.......?.......
027F70: EF 02 00 00 00 00 00 00  F0 02 00 00 00 00 00 00 | ?.......?.......
027F80: F1 02 00 00 00 00 00 00  F2 02 00 00 00 00 00 00 | ?.......?.......
027F90: F3 02 00 00 00 00 00 00  F4 02 00 00 00 00 00 00 | ?.......?.......
027FA0: F5 02 00 00 00 00 00 00  F6 02 00 00 00 00 00 00 | ?.......?.......
027FB0: F7 02 00 00 00 00 00 00  F8 02 00 00 00 00 00 00 | ?.......?.......
027FC0: F9 02 00 00 00 00 00 00  FA 02 00 00 00 00 00 00 | ?.......?.......
027FD0: FB 02 00 00 00 00 00 00  FC 02 00 00 00 00 00 00 | ?.......?.......
027FE0: FD 02 00 00 00 00 00 00  FE 02 00 00 00 00 00 00 | ?.......?.......
027FF0: FF 02 00 00 00 00 00 00  00 03 00 00 00 00 00 00 | ?...............
```

```
028000: 01 03 00 00 00 00 00 00  02 03 00 00 00 00 00 00 | ................
028010: 03 03 00 00 00 00 00 00  04 03 00 00 00 00 00 00 | ................
028020: 05 03 00 00 00 00 00 00  06 03 00 00 00 00 00 00 | ................
028030: 07 03 00 00 00 00 00 00  08 03 00 00 00 00 00 00 | ................
028040: 09 03 00 00 00 00 00 00  00 03 00 00 00 00 00 00 | ................
028050: 0B 03 00 00 00 00 00 00  0C 03 00 00 00 00 00 00 | ................
028060: 0D 03 00 00 00 00 00 00  0E 03 00 00 00 00 00 00 | ................
028070: 0F 03 00 00 00 00 00 00  10 03 00 00 00 00 00 00 | ................
028080: 11 03 00 00 00 00 00 00  12 03 00 00 00 00 00 00 | ................
028090: 13 03 00 00 00 00 00 00  14 03 00 00 00 00 00 00 | ................
0280A0: 15 03 00 00 00 00 00 00  16 03 00 00 00 00 00 00 | ................
0280B0: 17 03 00 00 00 00 00 00  18 03 00 00 00 00 00 00 | ................
0280C0: 19 03 00 00 00 00 00 00  1A 03 00 00 00 00 00 00 | ................
0280D0: 1B 03 00 00 00 00 00 00  1C 03 00 00 00 00 00 00 | ................
0280E0: 1D 03 00 00 00 00 00 00  1E 03 00 00 00 00 00 00 | ................
0280F0: 1F 03 00 00 00 00 00 00  20 03 00 00 00 00 00 00 | ........ .......

028100: 21 03 00 00 00 00 00 00  22 03 00 00 00 00 00 00 | !.......".......
028110: 23 03 00 00 00 00 00 00  24 03 00 00 00 00 00 00 | #.......$.......
028120: 25 03 00 00 00 00 00 00  26 03 00 00 00 00 00 00 | %.......&.......
028130: 27 03 00 00 00 00 00 00  28 03 00 00 00 00 00 00 | '.......(.......
028140: 29 03 00 00 00 00 00 00  2A 03 00 00 00 00 00 00 | ).......*.......
028150: 2B 03 00 00 00 00 00 00  2C 03 00 00 00 00 00 00 | +.......,.......
028160: 2D 03 00 00 00 00 00 00  2E 03 00 00 00 00 00 00 | -...............
028170: 00 03 00 00 00 00 00 00  30 03 00 00 00 00 00 00 | ........0.......
028180: 31 03 00 00 00 00 00 00  32 03 00 00 00 00 00 00 | 1.......2.......
028190: 33 03 00 00 00 00 00 00  34 03 00 00 00 00 00 00 | 3.......4.......
0281A0: 35 03 00 00 00 00 00 00  36 03 00 00 00 00 00 00 | 5.......6.......
0281B0: 37 03 00 00 00 00 00 00  38 03 00 00 00 00 00 00 | 7.......8.......
0281C0: 39 03 00 00 00 00 00 00  3A 03 00 00 00 00 00 00 | 9.......:.......
0281D0: 3B 03 00 00 00 00 00 00  3C 03 00 00 00 00 00 00 | ;.......<.......
0281E0: 3D 03 00 00 00 00 00 00  3E 03 00 00 00 00 00 00 | =.......>.......
0281F0: 3F 03 00 00 00 00 00 00  40 03 00 00 00 00 00 00 | ?.......@.......
```

```
028200: 41 03 00 00 00 00 00 00  42 03 00 00 00 00 00 00 | A.......B.......
028210: 43 03 00 00 00 00 00 00  44 03 00 00 00 00 00 00 | C.......D.......
028220: 45 03 00 00 00 00 00 00  46 03 00 00 00 00 00 00 | E.......F.......
028230: 47 03 00 00 00 00 00 00  48 03 00 00 00 00 00 00 | G.......H.......
028240: 49 03 00 00 00 00 00 00  4A 03 00 00 00 00 00 00 | I.......J.......
028250: 4B 03 00 00 00 00 00 00  4C 03 00 00 00 00 00 00 | K.......L.......
028260: 4D 03 00 00 00 00 00 00  4E 03 00 00 00 00 00 00 | M.......N.......
028270: 4F 03 00 00 00 00 00 00  50 03 00 00 00 00 00 00 | O.......P.......
028280: 51 03 00 00 00 00 00 00  52 03 00 00 00 00 00 00 | Q.......R.......
028290: 53 03 00 00 00 00 00 00  00 03 00 00 00 00 00 00 | S...............
0282A0: 55 03 00 00 00 00 00 00  56 03 00 00 00 00 00 00 | U.......V.......
0282B0: 57 03 00 00 00 00 00 00  58 03 00 00 00 00 00 00 | W.......X.......
0282C0: 59 03 00 00 00 00 00 00  5A 03 00 00 00 00 00 00 | Y.......Z.......
0282D0: 5B 03 00 00 00 00 00 00  5C 03 00 00 00 00 00 00 | [.......\.......
0282E0: 5D 03 00 00 00 00 00 00  5E 03 00 00 00 00 00 00 | ].......^.......
0282F0: 5F 03 00 00 00 00 00 00  60 03 00 00 00 00 00 00 | _.......`.......

028300: 61 03 00 00 00 00 00 00  62 03 00 00 00 00 00 00 | a.......b.......
028310: 63 03 00 00 00 00 00 00  64 03 00 00 00 00 00 00 | c.......d.......
028320: 65 03 00 00 00 00 00 00  66 03 00 00 00 00 00 00 | e.......f.......
028330: 67 03 00 00 00 00 00 00  68 03 00 00 00 00 00 00 | g.......h.......
028340: 69 03 00 00 00 00 00 00  6A 03 00 00 00 00 00 00 | i.......j.......
028350: 6B 03 00 00 00 00 00 00  6C 03 00 00 00 00 00 00 | k.......l.......
028360: 6D 03 00 00 00 00 00 00  6E 03 00 00 00 00 00 00 | m.......n.......
028370: 6F 03 00 00 00 00 00 00  70 03 00 00 00 00 00 00 | o.......p.......
028380: 71 03 00 00 00 00 00 00  72 03 00 00 00 00 00 00 | q.......r.......
028390: 73 03 00 00 00 00 00 00  74 03 00 00 00 00 00 00 | s.......t.......
0283A0: 75 03 00 00 00 00 00 00  76 03 00 00 00 00 00 00 | u.......v.......
0283B0: 77 03 00 00 00 00 00 00  78 03 00 00 00 00 00 00 | w.......x.......
0283C0: 00 03 00 00 00 00 00 00  7A 03 00 00 00 00 00 00 | ........z.......
0283D0: 7B 03 00 00 00 00 00 00  7C 03 00 00 00 00 00 00 | {.......|.......
0283E0: 7D 03 00 00 00 00 00 00  7E 03 00 00 00 00 00 00 | }.......~.......
0283F0: 7F 03 00 00 00 00 00 00  80 03 00 00 00 00 00 00 | .......◆.......
```

```
028400: 81 03 00 00 00 00 00 00  82 03 00 00 00 00 00 00 | ....................
028410: 83 03 00 00 00 00 00 00  84 03 00 00 00 00 00 00 | ....................
028420: 85 03 00 00 00 00 00 00  86 03 00 00 00 00 00 00 | ....................
028430: 87 03 00 00 00 00 00 00  88 03 00 00 00 00 00 00 | ....................
028440: 89 03 00 00 00 00 00 00  8A 03 00 00 00 00 00 00 | ....................
028450: 8B 03 00 00 00 00 00 00  8C 03 00 00 00 00 00 00 | ....................
028460: 8D 03 00 00 00 00 00 00  8E 03 00 00 00 00 00 00 | ....................
028470: 8F 03 00 00 00 00 00 00  90 03 00 00 00 00 00 00 | ....................
028480: 91 03 00 00 00 00 00 00  92 03 00 00 00 00 00 00 | ....................
028490: 93 03 00 00 00 00 00 00  94 03 00 00 00 00 00 00 | ....................
0284A0: 95 03 00 00 00 00 00 00  96 03 00 00 00 00 00 00 | ....................
0284B0: 97 03 00 00 00 00 00 00  98 03 00 00 00 00 00 00 | ....................
0284C0: 99 03 00 00 00 00 00 00  9A 03 00 00 00 00 00 00 | ....................
0284D0: 9B 03 00 00 00 00 00 00  9C 03 00 00 00 00 00 00 | ....................
0284E0: 9D 03 00 00 00 00 00 00  00 03 00 00 00 00 00 00 | ....................
0284F0: 9F 03 00 00 00 00 00 00  A0 03 00 00 00 00 00 00 | ....................

028500: A1 03 00 00 00 00 00 00  A2 03 00 00 00 00 00 00 | ....................
028510: A3 03 00 00 00 00 00 00  A4 03 00 00 00 00 00 00 | ....................
028520: A5 03 00 00 00 00 00 00  A6 03 00 00 00 00 00 00 | ....................
028530: A7 03 00 00 00 00 00 00  A8 03 00 00 00 00 00 00 | ....................
028540: A9 03 00 00 00 00 00 00  AA 03 00 00 00 00 00 00 | ....................
028550: AB 03 00 00 00 00 00 00  AC 03 00 00 00 00 00 00 | ....................
028560: AD 03 00 00 00 00 00 00  AE 03 00 00 00 00 00 00 | ....................
028570: AF 03 00 00 00 00 00 00  B0 03 00 00 00 00 00 00 | ....................
028580: B1 03 00 00 00 00 00 00  B2 03 00 00 00 00 00 00 | ....................
028590: B3 03 00 00 00 00 00 00  B4 03 00 00 00 00 00 00 | ....................
0285A0: B5 03 00 00 00 00 00 00  B6 03 00 00 00 00 00 00 | ....................
0285B0: B7 03 00 00 00 00 00 00  B8 03 00 00 00 00 00 00 | ....................
0285C0: B9 03 00 00 00 00 00 00  BA 03 00 00 00 00 00 00 | ....................
0285D0: BB 03 00 00 00 00 00 00  BC 03 00 00 00 00 00 00 | ....................
0285E0: BD 03 00 00 00 00 00 00  BE 03 00 00 00 00 00 00 | ....................
0285F0: BF 03 00 00 00 00 00 00  C0 03 00 00 00 00 00 00 | ....................
```

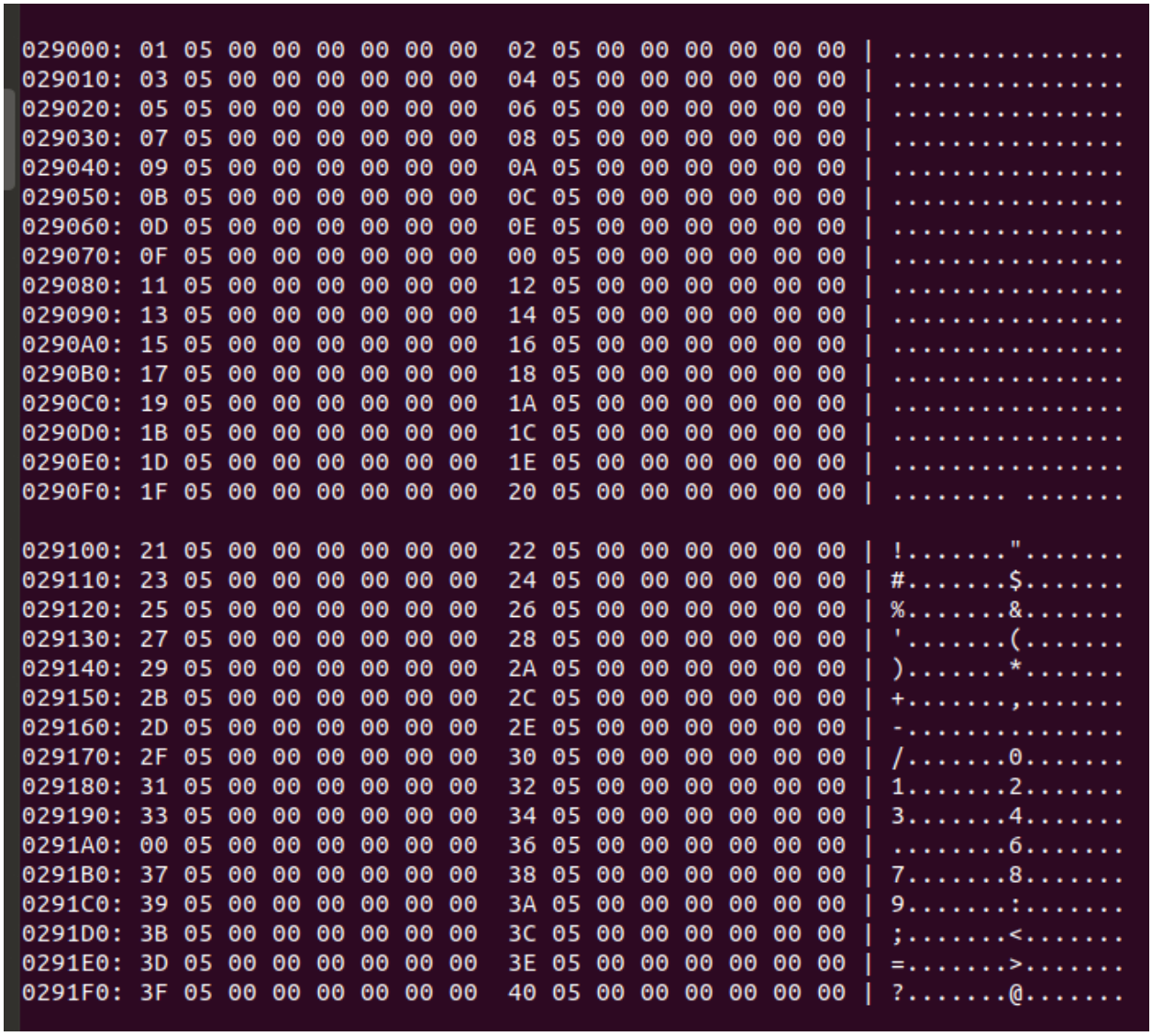
```
028600: C1 03 00 00 00 00 00 00  C2 03 00 00 00 00 00 00 | ◆.......◆.......
028610: 00 03 00 00 00 00 00 00  C4 03 00 00 00 00 00 00 | ........◆.......
028620: C5 03 00 00 00 00 00 00  C6 03 00 00 00 00 00 00 | ◆.......◆.......
028630: C7 03 00 00 00 00 00 00  C8 03 00 00 00 00 00 00 | ◆.......◆.......
028640: C9 03 00 00 00 00 00 00  CA 03 00 00 00 00 00 00 | ◆.......◆.......
028650: CB 03 00 00 00 00 00 00  CC 03 00 00 00 00 00 00 | ◆.......◆.......
028660: CD 03 00 00 00 00 00 00  CE 03 00 00 00 00 00 00 | ◆.......◆.......
028670: CF 03 00 00 00 00 00 00  D0 03 00 00 00 00 00 00 | ◆.......◆.......
028680: D1 03 00 00 00 00 00 00  D2 03 00 00 00 00 00 00 | ◆.......◆.......
028690: D3 03 00 00 00 00 00 00  D4 03 00 00 00 00 00 00 | ◆.......◆.......
0286A0: D5 03 00 00 00 00 00 00  D6 03 00 00 00 00 00 00 | ◆.......◆.......
0286B0: D7 03 00 00 00 00 00 00  D8 03 00 00 00 00 00 00 | ◆.......◆.......
0286C0: D9 03 00 00 00 00 00 00  DA 03 00 00 00 00 00 00 | ◆.......◆.......
0286D0: DB 03 00 00 00 00 00 00  DC 03 00 00 00 00 00 00 | ◆.......◆.......
0286E0: DD 03 00 00 00 00 00 00  DE 03 00 00 00 00 00 00 | ◆.......◆.......
0286F0: DF 03 00 00 00 00 00 00  E0 03 00 00 00 00 00 00 | ◆.......◆.......

028700: E1 03 00 00 00 00 00 00  E2 03 00 00 00 00 00 00 | ◆.......◆.......
028710: E3 03 00 00 00 00 00 00  E4 03 00 00 00 00 00 00 | ◆.......◆.......
028720: E5 03 00 00 00 00 00 00  E6 03 00 00 00 00 00 00 | ◆.......◆.......
028730: E7 03 00 00 00 00 00 00  00 03 00 00 00 00 00 00 | ◆...............
028740: E9 03 00 00 00 00 00 00  EA 03 00 00 00 00 00 00 | ◆.......◆.......
028750: EB 03 00 00 00 00 00 00  EC 03 00 00 00 00 00 00 | ◆.......◆.......
028760: ED 03 00 00 00 00 00 00  EE 03 00 00 00 00 00 00 | ◆.......◆.......
028770: EF 03 00 00 00 00 00 00  F0 03 00 00 00 00 00 00 | ◆.......◆.......
028780: F1 03 00 00 00 00 00 00  F2 03 00 00 00 00 00 00 | ◆.......◆.......
028790: F3 03 00 00 00 00 00 00  F4 03 00 00 00 00 00 00 | ◆.......◆.......
0287A0: F5 03 00 00 00 00 00 00  F6 03 00 00 00 00 00 00 | ◆.......◆.......
0287B0: F7 03 00 00 00 00 00 00  F8 03 00 00 00 00 00 00 | ◆.......◆.......
0287C0: F9 03 00 00 00 00 00 00  FA 03 00 00 00 00 00 00 | ◆.......◆.......
0287D0: FB 03 00 00 00 00 00 00  FC 03 00 00 00 00 00 00 | ◆.......◆.......
0287E0: FD 03 00 00 00 00 00 00  FE 03 00 00 00 00 00 00 | ◆.......◆.......
0287F0: FF 03 00 00 00 00 00 00  00 04 00 00 00 00 00 00 | ◆...............
```

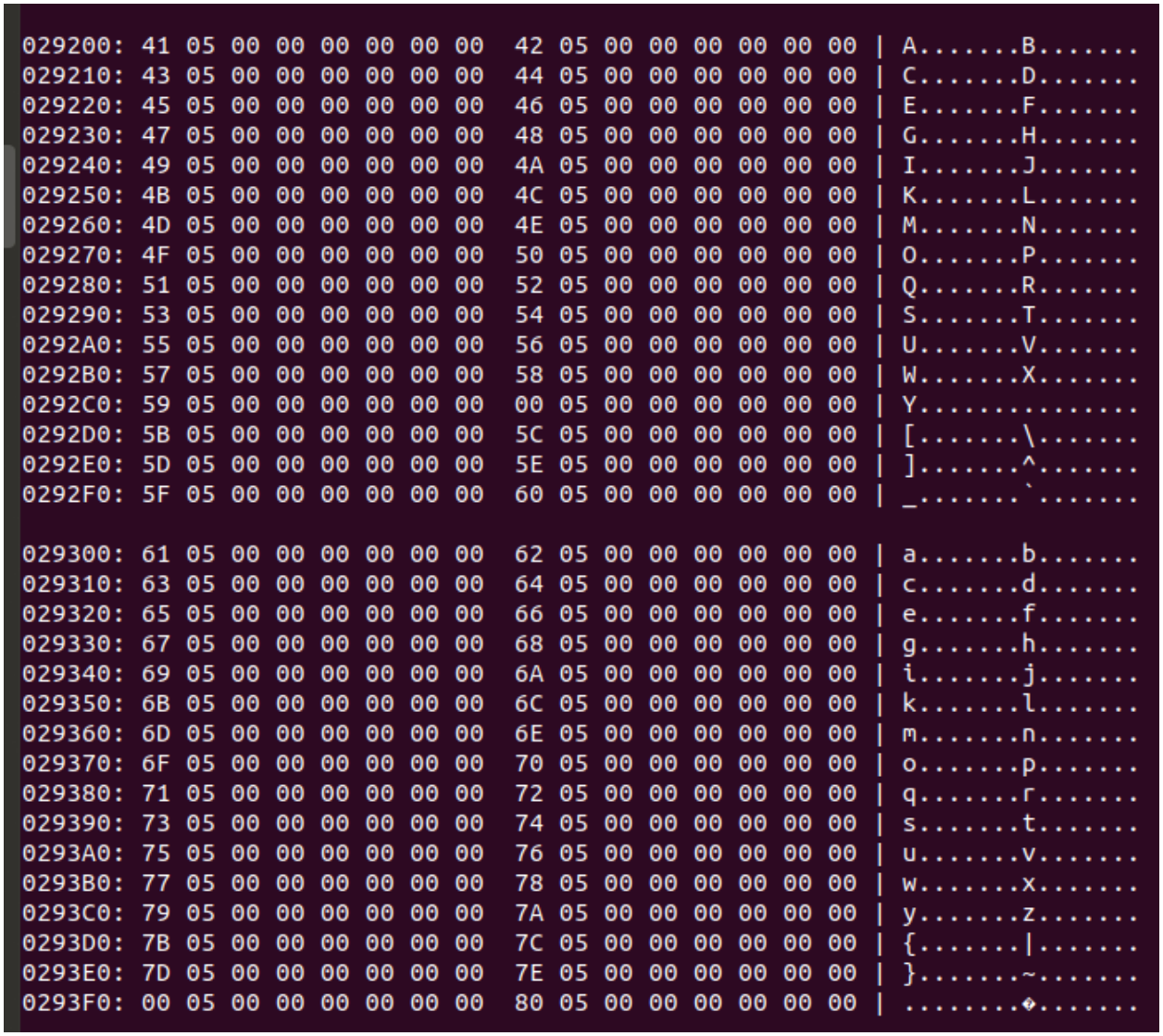
```
028800: 01 04 00 00 00 00 00 00  02 04 00 00 00 00 00 00 | ................
028810: 03 04 00 00 00 00 00 00  04 04 00 00 00 00 00 00 | ................
028820: 05 04 00 00 00 00 00 00  06 04 00 00 00 00 00 00 | ................
028830: 07 04 00 00 00 00 00 00  08 04 00 00 00 00 00 00 | ................
028840: 09 04 00 00 00 00 00 00  0A 04 00 00 00 00 00 00 | ................
028850: 0B 04 00 00 00 00 00 00  0C 04 00 00 00 00 00 00 | ................
028860: 00 04 00 00 00 00 00 00  0E 04 00 00 00 00 00 00 | ................
028870: 0F 04 00 00 00 00 00 00  10 04 00 00 00 00 00 00 | ................
028880: 11 04 00 00 00 00 00 00  12 04 00 00 00 00 00 00 | ................
028890: 13 04 00 00 00 00 00 00  14 04 00 00 00 00 00 00 | ................
0288A0: 15 04 00 00 00 00 00 00  16 04 00 00 00 00 00 00 | ................
0288B0: 17 04 00 00 00 00 00 00  18 04 00 00 00 00 00 00 | ................
0288C0: 19 04 00 00 00 00 00 00  1A 04 00 00 00 00 00 00 | ................
0288D0: 1B 04 00 00 00 00 00 00  1C 04 00 00 00 00 00 00 | ................
0288E0: 1D 04 00 00 00 00 00 00  1E 04 00 00 00 00 00 00 | ................
0288F0: 1F 04 00 00 00 00 00 00  20 04 00 00 00 00 00 00 | ........ .......

028900: 21 04 00 00 00 00 00 00  22 04 00 00 00 00 00 00 | !.......".......
028910: 23 04 00 00 00 00 00 00  24 04 00 00 00 00 00 00 | #.......$.......
028920: 25 04 00 00 00 00 00 00  26 04 00 00 00 00 00 00 | %.......&.......
028930: 27 04 00 00 00 00 00 00  28 04 00 00 00 00 00 00 | '.......(.......
028940: 29 04 00 00 00 00 00 00  2A 04 00 00 00 00 00 00 | ).......*.......
028950: 2B 04 00 00 00 00 00 00  2C 04 00 00 00 00 00 00 | +.......,.......
028960: 2D 04 00 00 00 00 00 00  2E 04 00 00 00 00 00 00 | -...............
028970: 2F 04 00 00 00 00 00 00  30 04 00 00 00 00 00 00 | /.......0.......
028980: 31 04 00 00 00 00 00 00  00 04 00 00 00 00 00 00 | 1...............
028990: 33 04 00 00 00 00 00 00  34 04 00 00 00 00 00 00 | 3.......4.......
0289A0: 35 04 00 00 00 00 00 00  36 04 00 00 00 00 00 00 | 5.......6.......
0289B0: 37 04 00 00 00 00 00 00  38 04 00 00 00 00 00 00 | 7.......8.......
0289C0: 39 04 00 00 00 00 00 00  3A 04 00 00 00 00 00 00 | 9.......:.......
0289D0: 3B 04 00 00 00 00 00 00  3C 04 00 00 00 00 00 00 | ;.......<.......
0289E0: 3D 04 00 00 00 00 00 00  3E 04 00 00 00 00 00 00 | =.......>.......
0289F0: 3F 04 00 00 00 00 00 00  40 04 00 00 00 00 00 00 | ?.......@.......
```

```
028A00: 41 04 00 00 00 00 00 00  42 04 00 00 00 00 00 00 | A.......B.......
028A10: 43 04 00 00 00 00 00 00  44 04 00 00 00 00 00 00 | C.......D.......
028A20: 45 04 00 00 00 00 00 00  46 04 00 00 00 00 00 00 | E.......F.......
028A30: 47 04 00 00 00 00 00 00  48 04 00 00 00 00 00 00 | G.......H.......
028A40: 49 04 00 00 00 00 00 00  4A 04 00 00 00 00 00 00 | I.......J.......
028A50: 4B 04 00 00 00 00 00 00  4C 04 00 00 00 00 00 00 | K.......L.......
028A60: 4D 04 00 00 00 00 00 00  4E 04 00 00 00 00 00 00 | M.......N.......
028A70: 4F 04 00 00 00 00 00 00  50 04 00 00 00 00 00 00 | O.......P.......
028A80: 51 04 00 00 00 00 00 00  52 04 00 00 00 00 00 00 | Q.......R.......
028A90: 53 04 00 00 00 00 00 00  54 04 00 00 00 00 00 00 | S.......T.......
028AA0: 55 04 00 00 00 00 00 00  56 04 00 00 00 00 00 00 | U.......V.......
028AB0: 00 04 00 00 00 00 00 00  58 04 00 00 00 00 00 00 | ........X.......
028AC0: 59 04 00 00 00 00 00 00  5A 04 00 00 00 00 00 00 | Y.......Z.......
028AD0: 5B 04 00 00 00 00 00 00  5C 04 00 00 00 00 00 00 | [.......\.......
028AE0: 5D 04 00 00 00 00 00 00  5E 04 00 00 00 00 00 00 | ].......^.......
028AF0: 5F 04 00 00 00 00 00 00  60 04 00 00 00 00 00 00 | _.......`.......

028B00: 61 04 00 00 00 00 00 00  62 04 00 00 00 00 00 00 | a.......b.......
028B10: 63 04 00 00 00 00 00 00  64 04 00 00 00 00 00 00 | c.......d.......
028B20: 65 04 00 00 00 00 00 00  66 04 00 00 00 00 00 00 | e.......f.......
028B30: 67 04 00 00 00 00 00 00  68 04 00 00 00 00 00 00 | g.......h.......
028B40: 69 04 00 00 00 00 00 00  6A 04 00 00 00 00 00 00 | i.......j.......
028B50: 6B 04 00 00 00 00 00 00  6C 04 00 00 00 00 00 00 | k.......l.......
028B60: 6D 04 00 00 00 00 00 00  6E 04 00 00 00 00 00 00 | m.......n.......
028B70: 6F 04 00 00 00 00 00 00  70 04 00 00 00 00 00 00 | o.......p.......
028B80: 71 04 00 00 00 00 00 00  72 04 00 00 00 00 00 00 | q.......r.......
028B90: 73 04 00 00 00 00 00 00  74 04 00 00 00 00 00 00 | s.......t.......
028BA0: 75 04 00 00 00 00 00 00  76 04 00 00 00 00 00 00 | u.......v.......
028BB0: 77 04 00 00 00 00 00 00  78 04 00 00 00 00 00 00 | w.......x.......
028BC0: 79 04 00 00 00 00 00 00  7A 04 00 00 00 00 00 00 | y.......z.......
028BD0: 7B 04 00 00 00 00 00 00  00 04 00 00 00 00 00 00 | {...............
028BE0: 7D 04 00 00 00 00 00 00  7E 04 00 00 00 00 00 00 | }.......~.......
028BF0: 7F 04 00 00 00 00 00 00  80 04 00 00 00 00 00 00 | .......?.......
```

```
028C00: 81 04 00 00 00 00 00 00  82 04 00 00 00 00 00 00 | ♦.......♦.......
028C10: 83 04 00 00 00 00 00 00  84 04 00 00 00 00 00 00 | ♦.......♦.......
028C20: 85 04 00 00 00 00 00 00  86 04 00 00 00 00 00 00 | ♦.......♦.......
028C30: 87 04 00 00 00 00 00 00  88 04 00 00 00 00 00 00 | ♦.......♦.......
028C40: 89 04 00 00 00 00 00 00  8A 04 00 00 00 00 00 00 | ♦.......♦.......
028C50: 8B 04 00 00 00 00 00 00  8C 04 00 00 00 00 00 00 | ♦.......♦.......
028C60: 8D 04 00 00 00 00 00 00  8E 04 00 00 00 00 00 00 | ♦.......♦.......
028C70: 8F 04 00 00 00 00 00 00  90 04 00 00 00 00 00 00 | ♦.......♦.......
028C80: 91 04 00 00 00 00 00 00  92 04 00 00 00 00 00 00 | ♦.......♦.......
028C90: 93 04 00 00 00 00 00 00  94 04 00 00 00 00 00 00 | ♦.......♦.......
028CA0: 95 04 00 00 00 00 00 00  96 04 00 00 00 00 00 00 | ♦.......♦.......
028CB0: 97 04 00 00 00 00 00 00  98 04 00 00 00 00 00 00 | ♦.......♦.......
028CC0: 99 04 00 00 00 00 00 00  9A 04 00 00 00 00 00 00 | ♦.......♦.......
028CD0: 9B 04 00 00 00 00 00 00  9C 04 00 00 00 00 00 00 | ♦.......♦.......
028CE0: 9D 04 00 00 00 00 00 00  9E 04 00 00 00 00 00 00 | ♦.......♦.......
028CF0: 9F 04 00 00 00 00 00 00  A0 04 00 00 00 00 00 00 | ♦.......♦.......

028D00: 00 04 00 00 00 00 00 00  A2 04 00 00 00 00 00 00 | ........♦.......
028D10: A3 04 00 00 00 00 00 00  A4 04 00 00 00 00 00 00 | ♦.......♦.......
028D20: A5 04 00 00 00 00 00 00  A6 04 00 00 00 00 00 00 | ♦.......♦.......
028D30: A7 04 00 00 00 00 00 00  A8 04 00 00 00 00 00 00 | ♦.......♦.......
028D40: A9 04 00 00 00 00 00 00  AA 04 00 00 00 00 00 00 | ♦.......♦.......
028D50: AB 04 00 00 00 00 00 00  AC 04 00 00 00 00 00 00 | ♦.......♦.......
028D60: AD 04 00 00 00 00 00 00  AE 04 00 00 00 00 00 00 | ♦.......♦.......
028D70: AF 04 00 00 00 00 00 00  B0 04 00 00 00 00 00 00 | ♦.......♦.......
028D80: B1 04 00 00 00 00 00 00  B2 04 00 00 00 00 00 00 | ♦.......♦.......
028D90: B3 04 00 00 00 00 00 00  B4 04 00 00 00 00 00 00 | ♦.......♦.......
028DA0: B5 04 00 00 00 00 00 00  B6 04 00 00 00 00 00 00 | ♦.......♦.......
028DB0: B7 04 00 00 00 00 00 00  B8 04 00 00 00 00 00 00 | ♦.......♦.......
028DC0: B9 04 00 00 00 00 00 00  BA 04 00 00 00 00 00 00 | ♦.......♦.......
028DD0: BB 04 00 00 00 00 00 00  BC 04 00 00 00 00 00 00 | ♦.......♦.......
028DE0: BD 04 00 00 00 00 00 00  BE 04 00 00 00 00 00 00 | ♦.......♦.......
028DF0: BF 04 00 00 00 00 00 00  C0 04 00 00 00 00 00 00 | ♦.......♦.......
```

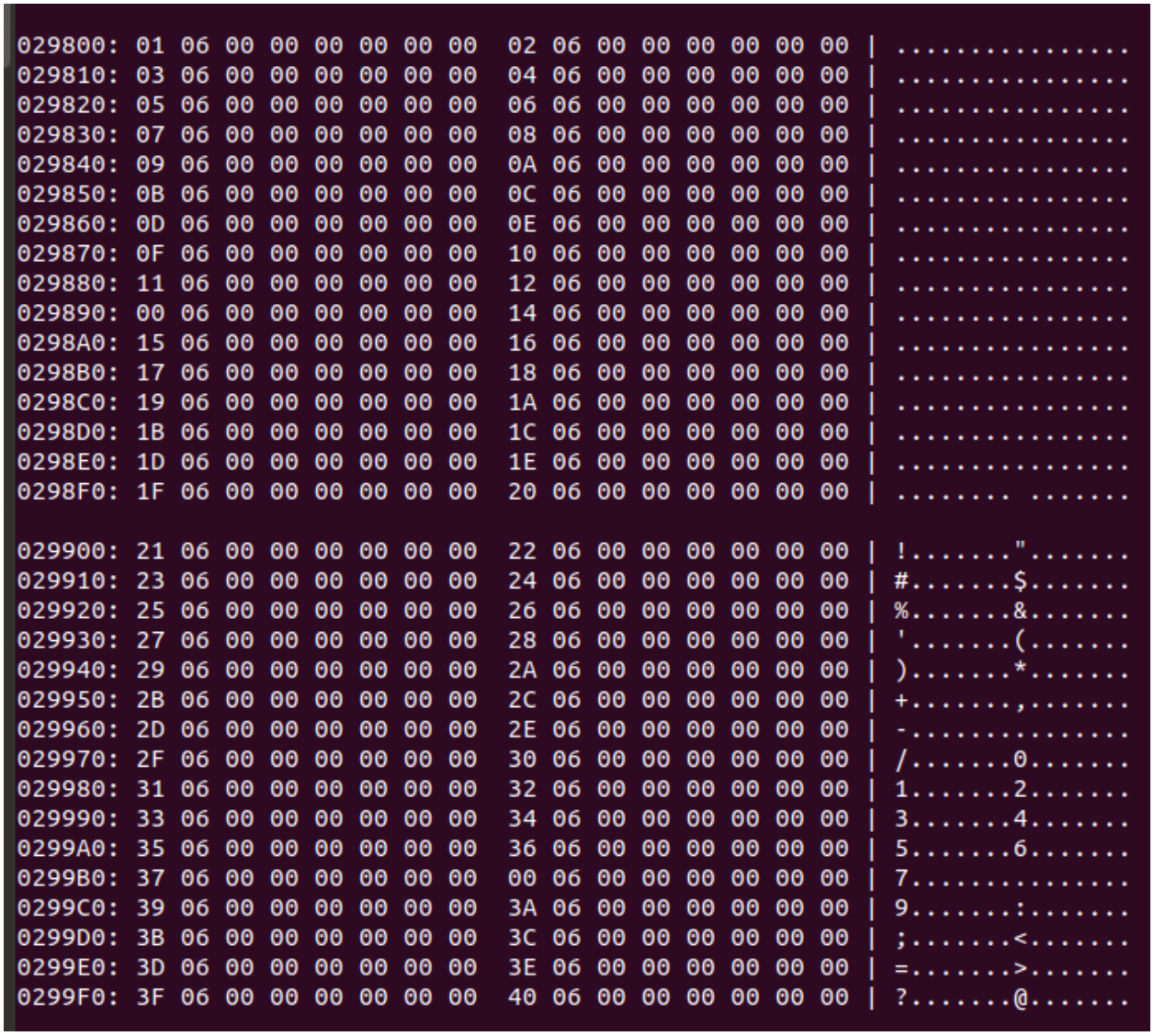
```
028E00: C1 04 00 00 00 00 00 00  C2 04 00 00 00 00 00 00 | ◆.......◆.......
028E10: C3 04 00 00 00 00 00 00  C4 04 00 00 00 00 00 00 | ◆.......◆.......
028E20: C5 04 00 00 00 00 00 00  00 04 00 00 00 00 00 00 | ◆...............
028E30: C7 04 00 00 00 00 00 00  C8 04 00 00 00 00 00 00 | ◆.......◆.......
028E40: C9 04 00 00 00 00 00 00  CA 04 00 00 00 00 00 00 | ◆.......◆.......
028E50: CB 04 00 00 00 00 00 00  CC 04 00 00 00 00 00 00 | ◆.......◆.......
028E60: CD 04 00 00 00 00 00 00  CE 04 00 00 00 00 00 00 | ◆.......◆.......
028E70: CF 04 00 00 00 00 00 00  D0 04 00 00 00 00 00 00 | ◆.......◆.......
028E80: D1 04 00 00 00 00 00 00  D2 04 00 00 00 00 00 00 | ◆.......◆.......
028E90: D3 04 00 00 00 00 00 00  D4 04 00 00 00 00 00 00 | ◆.......◆.......
028EA0: D5 04 00 00 00 00 00 00  D6 04 00 00 00 00 00 00 | ◆.......◆.......
028EB0: D7 04 00 00 00 00 00 00  D8 04 00 00 00 00 00 00 | ◆.......◆.......
028EC0: D9 04 00 00 00 00 00 00  DA 04 00 00 00 00 00 00 | ◆.......◆.......
028ED0: DB 04 00 00 00 00 00 00  DC 04 00 00 00 00 00 00 | ◆.......◆.......
028EE0: DD 04 00 00 00 00 00 00  DE 04 00 00 00 00 00 00 | ◆.......◆.......
028EF0: DF 04 00 00 00 00 00 00  E0 04 00 00 00 00 00 00 | ◆.......◆.......

028F00: E1 04 00 00 00 00 00 00  E2 04 00 00 00 00 00 00 | ◆.......◆.......
028F10: E3 04 00 00 00 00 00 00  E4 04 00 00 00 00 00 00 | ◆.......◆.......
028F20: E5 04 00 00 00 00 00 00  E6 04 00 00 00 00 00 00 | ◆.......◆.......
028F30: E7 04 00 00 00 00 00 00  E8 04 00 00 00 00 00 00 | ◆.......◆.......
028F40: E9 04 00 00 00 00 00 00  EA 04 00 00 00 00 00 00 | ◆.......◆.......
028F50: 00 04 00 00 00 00 00 00  EC 04 00 00 00 00 00 00 | ........◆.......
028F60: ED 04 00 00 00 00 00 00  EE 04 00 00 00 00 00 00 | ◆.......◆.......
028F70: EF 04 00 00 00 00 00 00  F0 04 00 00 00 00 00 00 | ◆.......◆.......
028F80: F1 04 00 00 00 00 00 00  F2 04 00 00 00 00 00 00 | ◆.......◆.......
028F90: F3 04 00 00 00 00 00 00  F4 04 00 00 00 00 00 00 | ◆.......◆.......
028FA0: F5 04 00 00 00 00 00 00  F6 04 00 00 00 00 00 00 | ◆.......◆.......
028FB0: F7 04 00 00 00 00 00 00  F8 04 00 00 00 00 00 00 | ◆.......◆.......
028FC0: F9 04 00 00 00 00 00 00  FA 04 00 00 00 00 00 00 | ◆.......◆.......
028FD0: FB 04 00 00 00 00 00 00  FC 04 00 00 00 00 00 00 | ◆.......◆.......
028FE0: FD 04 00 00 00 00 00 00  FE 04 00 00 00 00 00 00 | ◆.......◆.......
028FF0: FF 04 00 00 00 00 00 00  00 05 00 00 00 00 00 00 | ◆...............
```


029000: 01 05 00 00 00 00 00 00 02 05 00 00 00 00 00 00 | ....
029010: 03 05 00 00 00 00 00 00 04 05 00 00 00 00 00 00 | ....
029020: 05 05 00 00 00 00 00 00 06 05 00 00 00 00 00 00 | ....
029030: 07 05 00 00 00 00 00 00 08 05 00 00 00 00 00 00 | ....
029040: 09 05 00 00 00 00 00 00 0A 05 00 00 00 00 00 00 | ....
029050: 0B 05 00 00 00 00 00 00 0C 05 00 00 00 00 00 00 | ....
029060: 0D 05 00 00 00 00 00 00 0E 05 00 00 00 00 00 00 | ....
029070: 0F 05 00 00 00 00 00 00 00 05 00 00 00 00 00 00 | ....
029080: 11 05 00 00 00 00 00 00 12 05 00 00 00 00 00 00 | ....
029090: 13 05 00 00 00 00 00 00 14 05 00 00 00 00 00 00 | ....
0290A0: 15 05 00 00 00 00 00 00 16 05 00 00 00 00 00 00 | ....
0290B0: 17 05 00 00 00 00 00 00 18 05 00 00 00 00 00 00 | ....
0290C0: 19 05 00 00 00 00 00 00 1A 05 00 00 00 00 00 00 | ....
0290D0: 1B 05 00 00 00 00 00 00 1C 05 00 00 00 00 00 00 | ....
0290E0: 1D 05 00 00 00 00 00 00 1E 05 00 00 00 00 00 00 | ....
0290F0: 1F 05 00 00 00 00 00 00 20 05 00 00 00 00 00 00 | ....
029100: 21 05 00 00 00 00 00 00 22 05 00 00 00 00 00 00 | !.......".......
029110: 23 05 00 00 00 00 00 00 24 05 00 00 00 00 00 00 | #.......$.......
029120: 25 05 00 00 00 00 00 00 26 05 00 00 00 00 00 00 | %.......&.......
029130: 27 05 00 00 00 00 00 00 28 05 00 00 00 00 00 00 | '.......(.......
029140: 29 05 00 00 00 00 00 00 2A 05 00 00 00 00 00 00 |).......*.......
029150: 2B 05 00 00 00 00 00 00 2C 05 00 00 00 00 00 00 | +.......,.......
029160: 2D 05 00 00 00 00 00 00 2E 05 00 00 00 00 00 00 | -................
029170: 2F 05 00 00 00 00 00 00 30 05 00 00 00 00 00 00 | /.......0.......
029180: 31 05 00 00 00 00 00 00 32 05 00 00 00 00 00 00 | 1.......2.......
029190: 33 05 00 00 00 00 00 00 34 05 00 00 00 00 00 00 | 3.......4.......
0291A0: 00 05 00 00 00 00 00 00 36 05 00 00 00 00 00 00 | ....6.......
0291B0: 37 05 00 00 00 00 00 00 38 05 00 00 00 00 00 00 | 7.......8.......
0291C0: 39 05 00 00 00 00 00 00 3A 05 00 00 00 00 00 00 | 9.......:.......
0291D0: 3B 05 00 00 00 00 00 00 3C 05 00 00 00 00 00 00 | ;.......<.......
0291E0: 3D 05 00 00 00 00 00 00 3E 05 00 00 00 00 00 00 | =.......>.......
0291F0: 3F 05 00 00 00 00 00 00 40 05 00 00 00 00 00 00 | ?.......@.......

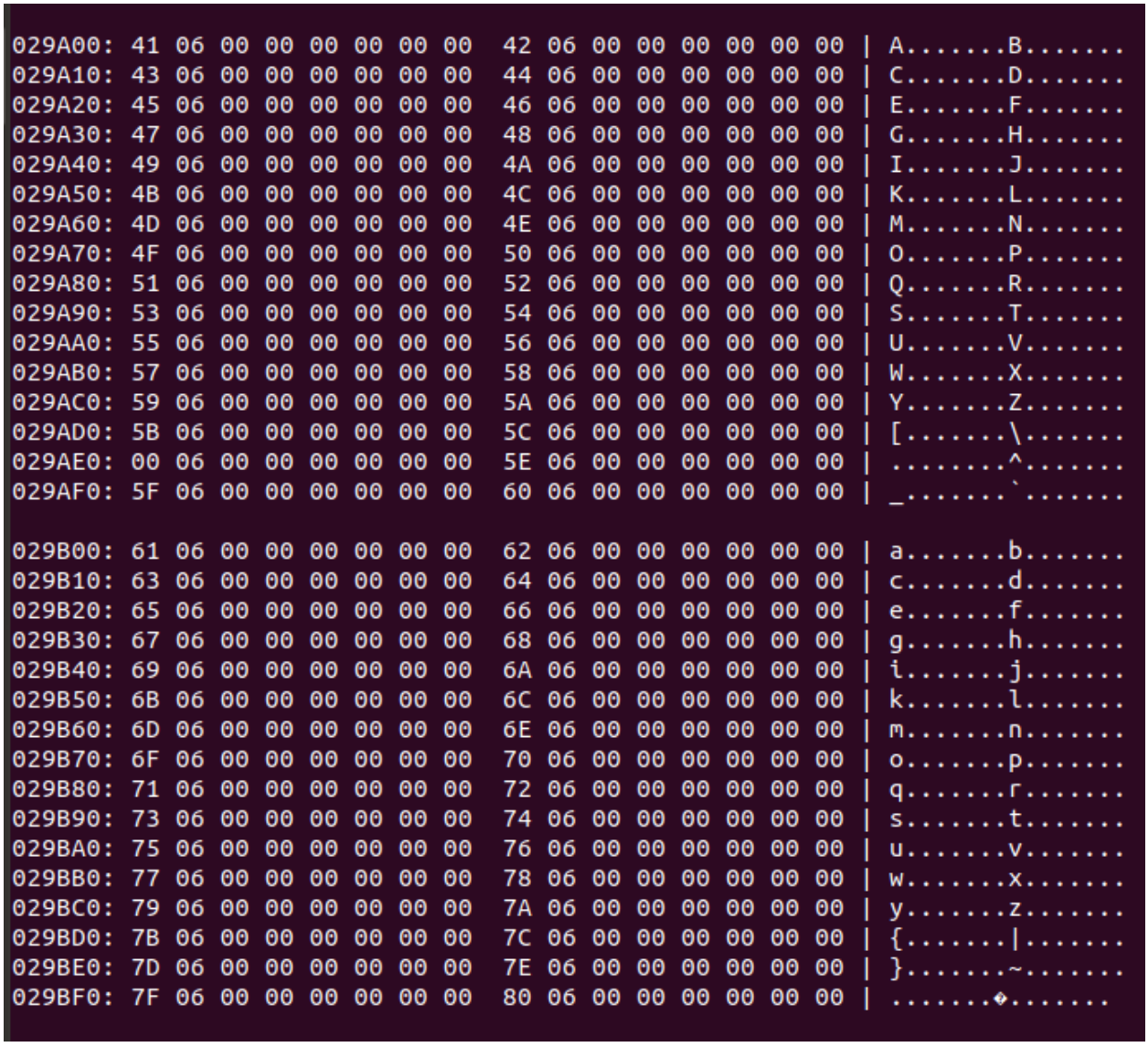

029200: 41 05 00 00 00 00 00 00 42 05 00 00 00 00 00 00 | A.......B.......
029210: 43 05 00 00 00 00 00 00 44 05 00 00 00 00 00 00 | C.......D.......
029220: 45 05 00 00 00 00 00 00 46 05 00 00 00 00 00 00 | E.......F.......
029230: 47 05 00 00 00 00 00 00 48 05 00 00 00 00 00 00 | G.......H.......
029240: 49 05 00 00 00 00 00 00 4A 05 00 00 00 00 00 00 | I.......J.......
029250: 4B 05 00 00 00 00 00 00 4C 05 00 00 00 00 00 00 | K.......L.......
029260: 4D 05 00 00 00 00 00 00 4E 05 00 00 00 00 00 00 | M.......N.......
029270: 4F 05 00 00 00 00 00 00 50 05 00 00 00 00 00 00 | O.......P.......
029280: 51 05 00 00 00 00 00 00 52 05 00 00 00 00 00 00 | Q.......R.......
029290: 53 05 00 00 00 00 00 00 54 05 00 00 00 00 00 00 | S.......T.......
0292A0: 55 05 00 00 00 00 00 00 56 05 00 00 00 00 00 00 | U.......V.......
0292B0: 57 05 00 00 00 00 00 00 58 05 00 00 00 00 00 00 | W.......X.......
0292C0: 59 05 00 00 00 00 00 00 00 05 00 00 00 00 00 00 | Y...............
0292D0: 5B 05 00 00 00 00 00 00 5C 05 00 00 00 00 00 00 | [.......\.......
0292E0: 5D 05 00 00 00 00 00 00 5E 05 00 00 00 00 00 00 |].......^.......
0292F0: 5F 05 00 00 00 00 00 00 60 05 00 00 00 00 00 00 | _.......`.......
029300: 61 05 00 00 00 00 00 00 62 05 00 00 00 00 00 00 | a.......b.......
029310: 63 05 00 00 00 00 00 00 64 05 00 00 00 00 00 00 | c.......d.......
029320: 65 05 00 00 00 00 00 00 66 05 00 00 00 00 00 00 | e.......f.......
029330: 67 05 00 00 00 00 00 00 68 05 00 00 00 00 00 00 | g.......h.......
029340: 69 05 00 00 00 00 00 00 6A 05 00 00 00 00 00 00 | i.......j.......
029350: 6B 05 00 00 00 00 00 00 6C 05 00 00 00 00 00 00 | k.......l.......
029360: 6D 05 00 00 00 00 00 00 6E 05 00 00 00 00 00 00 | m.......n.......
029370: 6F 05 00 00 00 00 00 00 70 05 00 00 00 00 00 00 | o.......p.......
029380: 71 05 00 00 00 00 00 00 72 05 00 00 00 00 00 00 | q.......r.......
029390: 73 05 00 00 00 00 00 00 74 05 00 00 00 00 00 00 | s.......t.......
0293A0: 75 05 00 00 00 00 00 00 76 05 00 00 00 00 00 00 | u.......v.......
0293B0: 77 05 00 00 00 00 00 00 78 05 00 00 00 00 00 00 | w.......x.......
0293C0: 79 05 00 00 00 00 00 00 7A 05 00 00 00 00 00 00 | y.......z.......
0293D0: 7B 05 00 00 00 00 00 00 7C 05 00 00 00 00 00 00 | {.......|.......
0293E0: 7D 05 00 00 00 00 00 00 7E 05 00 00 00 00 00 00 | }.......~.......
0293F0: 00 05 00 00 00 00 00 00 80 05 00 00 00 00 00 00 | ....◆.......

```
029400: 81 05 00 00 00 00 00 00  82 05 00 00 00 00 00 00 | ◆.......◆.......
029410: 83 05 00 00 00 00 00 00  84 05 00 00 00 00 00 00 | ◆.......◆.......
029420: 85 05 00 00 00 00 00 00  86 05 00 00 00 00 00 00 | ◆.......◆.......
029430: 87 05 00 00 00 00 00 00  88 05 00 00 00 00 00 00 | ◆.......◆.......
029440: 89 05 00 00 00 00 00 00  8A 05 00 00 00 00 00 00 | ◆.......◆.......
029450: 8B 05 00 00 00 00 00 00  8C 05 00 00 00 00 00 00 | ◆.......◆.......
029460: 8D 05 00 00 00 00 00 00  8E 05 00 00 00 00 00 00 | ◆.......◆.......
029470: 8F 05 00 00 00 00 00 00  90 05 00 00 00 00 00 00 | ◆.......◆.......
029480: 91 05 00 00 00 00 00 00  92 05 00 00 00 00 00 00 | ◆.......◆.......
029490: 93 05 00 00 00 00 00 00  94 05 00 00 00 00 00 00 | ◆.......◆.......
0294A0: 95 05 00 00 00 00 00 00  96 05 00 00 00 00 00 00 | ◆.......◆.......
0294B0: 97 05 00 00 00 00 00 00  98 05 00 00 00 00 00 00 | ◆.......◆.......
0294C0: 99 05 00 00 00 00 00 00  9A 05 00 00 00 00 00 00 | ◆.......◆.......
0294D0: 9B 05 00 00 00 00 00 00  9C 05 00 00 00 00 00 00 | ◆.......◆.......
0294E0: 9D 05 00 00 00 00 00 00  9E 05 00 00 00 00 00 00 | ◆.......◆.......
0294F0: 9F 05 00 00 00 00 00 00  A0 05 00 00 00 00 00 00 | ◆.......◆.......

029500: A1 05 00 00 00 00 00 00  A2 05 00 00 00 00 00 00 | ◆.......◆.......
029510: A3 05 00 00 00 00 00 00  00 05 00 00 00 00 00 00 | ◆...............
029520: A5 05 00 00 00 00 00 00  A6 05 00 00 00 00 00 00 | ◆.......◆.......
029530: A7 05 00 00 00 00 00 00  A8 05 00 00 00 00 00 00 | ◆.......◆.......
029540: A9 05 00 00 00 00 00 00  AA 05 00 00 00 00 00 00 | ◆.......◆.......
029550: AB 05 00 00 00 00 00 00  AC 05 00 00 00 00 00 00 | ◆.......◆.......
029560: AD 05 00 00 00 00 00 00  AE 05 00 00 00 00 00 00 | ◆.......◆.......
029570: AF 05 00 00 00 00 00 00  B0 05 00 00 00 00 00 00 | ◆.......◆.......
029580: B1 05 00 00 00 00 00 00  B2 05 00 00 00 00 00 00 | ◆.......◆.......
029590: B3 05 00 00 00 00 00 00  B4 05 00 00 00 00 00 00 | ◆.......◆.......
0295A0: B5 05 00 00 00 00 00 00  B6 05 00 00 00 00 00 00 | ◆.......◆.......
0295B0: B7 05 00 00 00 00 00 00  B8 05 00 00 00 00 00 00 | ◆.......◆.......
0295C0: B9 05 00 00 00 00 00 00  BA 05 00 00 00 00 00 00 | ◆.......◆.......
0295D0: BB 05 00 00 00 00 00 00  BC 05 00 00 00 00 00 00 | ◆.......◆.......
0295E0: BD 05 00 00 00 00 00 00  BE 05 00 00 00 00 00 00 | ◆.......◆.......
0295F0: BF 05 00 00 00 00 00 00  C0 05 00 00 00 00 00 00 | ◆.......◆.......
```

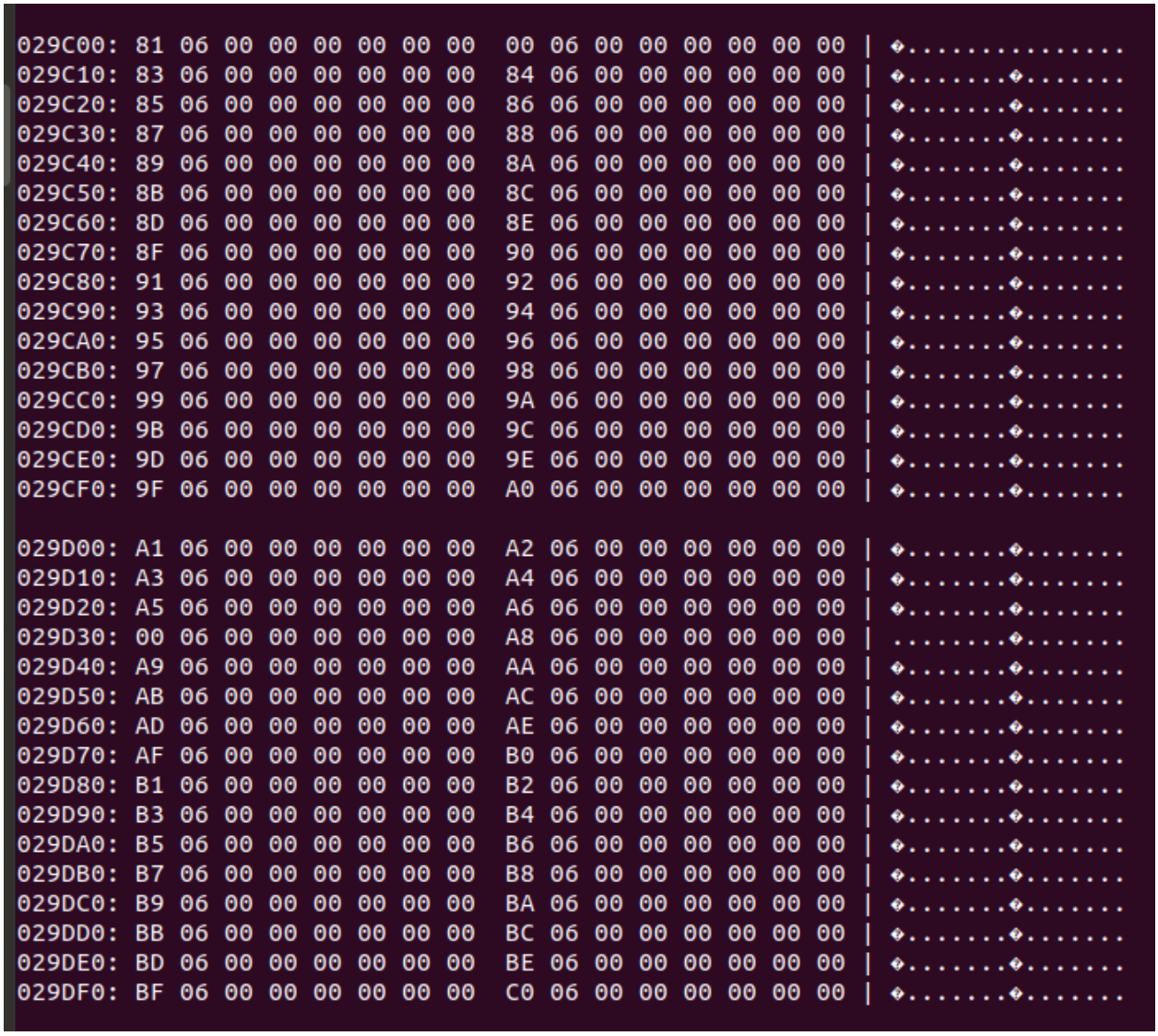
```
029600: C1 05 00 00 00 00 00 00  C2 05 00 00 00 00 00 00 | ?.......?.......
029610: C3 05 00 00 00 00 00 00  C4 05 00 00 00 00 00 00 | ?.......?.......
029620: C5 05 00 00 00 00 00 00  C6 05 00 00 00 00 00 00 | ?.......?.......
029630: C7 05 00 00 00 00 00 00  C8 05 00 00 00 00 00 00 | ?.......?.......
029640: 00 05 00 00 00 00 00 00  CA 05 00 00 00 00 00 00 |  .......?.......
029650: CB 05 00 00 00 00 00 00  CC 05 00 00 00 00 00 00 | ?.......?.......
029660: CD 05 00 00 00 00 00 00  CE 05 00 00 00 00 00 00 | ?.......?.......
029670: CF 05 00 00 00 00 00 00  D0 05 00 00 00 00 00 00 | ?.......?.......
029680: D1 05 00 00 00 00 00 00  D2 05 00 00 00 00 00 00 | ?.......?.......
029690: D3 05 00 00 00 00 00 00  D4 05 00 00 00 00 00 00 | ?.......?.......
0296A0: D5 05 00 00 00 00 00 00  D6 05 00 00 00 00 00 00 | ?.......?.......
0296B0: D7 05 00 00 00 00 00 00  D8 05 00 00 00 00 00 00 | ?.......?.......
0296C0: D9 05 00 00 00 00 00 00  DA 05 00 00 00 00 00 00 | ?.......?.......
0296D0: DB 05 00 00 00 00 00 00  DC 05 00 00 00 00 00 00 | ?.......?.......
0296E0: DD 05 00 00 00 00 00 00  DE 05 00 00 00 00 00 00 | ?.......?.......
0296F0: DF 05 00 00 00 00 00 00  E0 05 00 00 00 00 00 00 | ?.......?.......

029700: E1 05 00 00 00 00 00 00  E2 05 00 00 00 00 00 00 | ?.......?.......
029710: E3 05 00 00 00 00 00 00  E4 05 00 00 00 00 00 00 | ?.......?.......
029720: E5 05 00 00 00 00 00 00  E6 05 00 00 00 00 00 00 | ?.......?.......
029730: E7 05 00 00 00 00 00 00  E8 05 00 00 00 00 00 00 | ?.......?.......
029740: E9 05 00 00 00 00 00 00  EA 05 00 00 00 00 00 00 | ?.......?.......
029750: EB 05 00 00 00 00 00 00  EC 05 00 00 00 00 00 00 | ?.......?.......
029760: ED 05 00 00 00 00 00 00  00 05 00 00 00 00 00 00 | ?...............
029770: EF 05 00 00 00 00 00 00  F0 05 00 00 00 00 00 00 | ?.......?.......
029780: F1 05 00 00 00 00 00 00  F2 05 00 00 00 00 00 00 | ?.......?.......
029790: F3 05 00 00 00 00 00 00  F4 05 00 00 00 00 00 00 | ?.......?.......
0297A0: F5 05 00 00 00 00 00 00  F6 05 00 00 00 00 00 00 | ?.......?.......
0297B0: F7 05 00 00 00 00 00 00  F8 05 00 00 00 00 00 00 | ?.......?.......
0297C0: F9 05 00 00 00 00 00 00  FA 05 00 00 00 00 00 00 | ?.......?.......
0297D0: FB 05 00 00 00 00 00 00  FC 05 00 00 00 00 00 00 | ?.......?.......
0297E0: FD 05 00 00 00 00 00 00  FE 05 00 00 00 00 00 00 | ?.......?.......
0297F0: FF 05 00 00 00 00 00 00  00 06 00 00 00 00 00 00 | ?...............
```



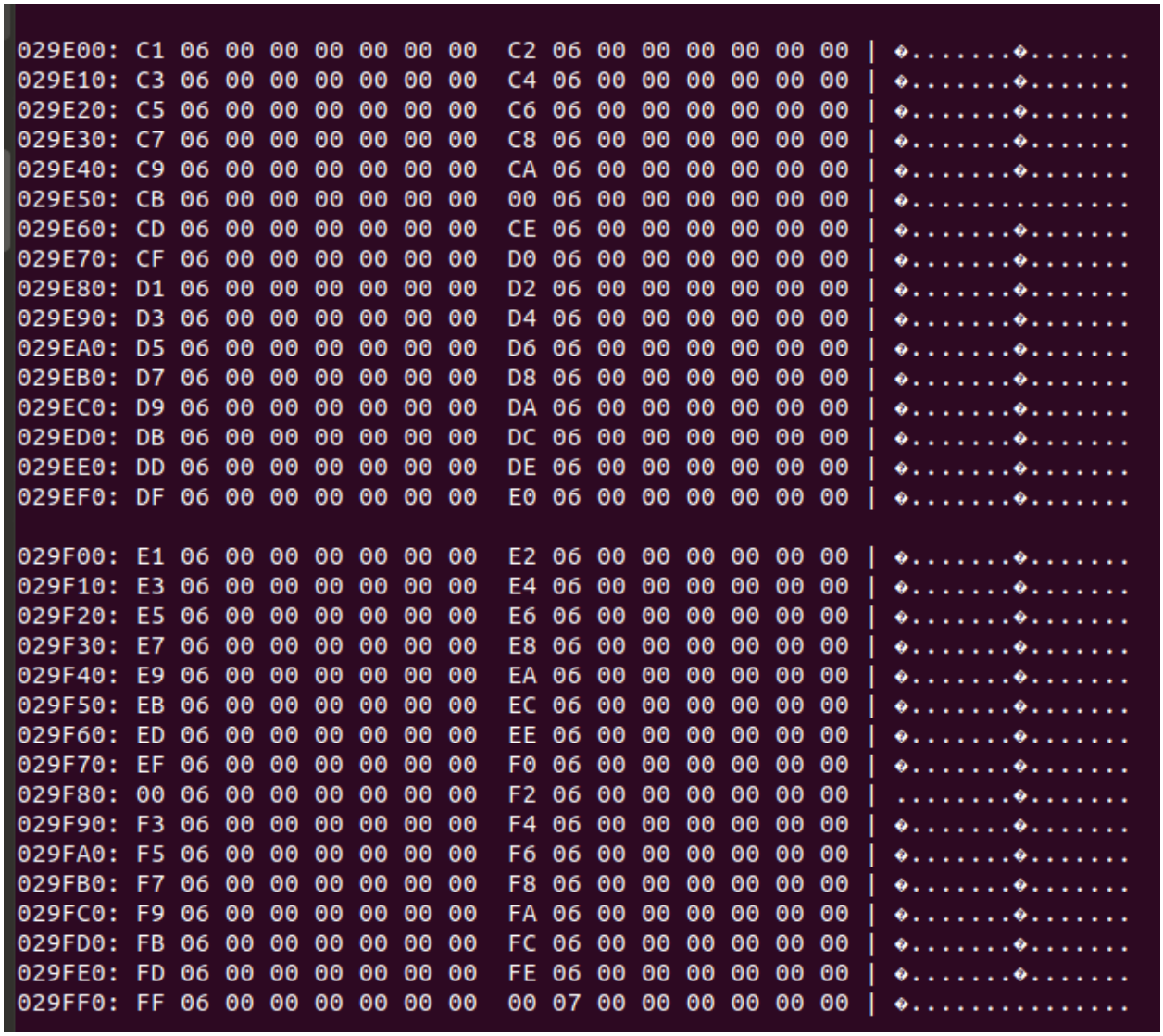
```
029800: 01 06 00 00 00 00 00 00  02 06 00 00 00 00 00 00 | ................
029810: 03 06 00 00 00 00 00 00  04 06 00 00 00 00 00 00 | ................
029820: 05 06 00 00 00 00 00 00  06 06 00 00 00 00 00 00 | ................
029830: 07 06 00 00 00 00 00 00  08 06 00 00 00 00 00 00 | ................
029840: 09 06 00 00 00 00 00 00  0A 06 00 00 00 00 00 00 | ................
029850: 0B 06 00 00 00 00 00 00  0C 06 00 00 00 00 00 00 | ................
029860: 0D 06 00 00 00 00 00 00  0E 06 00 00 00 00 00 00 | ................
029870: 0F 06 00 00 00 00 00 00  10 06 00 00 00 00 00 00 | ................
029880: 11 06 00 00 00 00 00 00  12 06 00 00 00 00 00 00 | ................
029890: 00 06 00 00 00 00 00 00  14 06 00 00 00 00 00 00 | ................
0298A0: 15 06 00 00 00 00 00 00  16 06 00 00 00 00 00 00 | ................
0298B0: 17 06 00 00 00 00 00 00  18 06 00 00 00 00 00 00 | ................
0298C0: 19 06 00 00 00 00 00 00  1A 06 00 00 00 00 00 00 | ................
0298D0: 1B 06 00 00 00 00 00 00  1C 06 00 00 00 00 00 00 | ................
0298E0: 1D 06 00 00 00 00 00 00  1E 06 00 00 00 00 00 00 | ................
0298F0: 1F 06 00 00 00 00 00 00  20 06 00 00 00 00 00 00 | ........ .......

029900: 21 06 00 00 00 00 00 00  22 06 00 00 00 00 00 00 | !.......".......
029910: 23 06 00 00 00 00 00 00  24 06 00 00 00 00 00 00 | #.......$.......
029920: 25 06 00 00 00 00 00 00  26 06 00 00 00 00 00 00 | %.......&.......
029930: 27 06 00 00 00 00 00 00  28 06 00 00 00 00 00 00 | '.......(.......
029940: 29 06 00 00 00 00 00 00  2A 06 00 00 00 00 00 00 | ).......*.......
029950: 2B 06 00 00 00 00 00 00  2C 06 00 00 00 00 00 00 | +.......,.......
029960: 2D 06 00 00 00 00 00 00  2E 06 00 00 00 00 00 00 | -................
029970: 2F 06 00 00 00 00 00 00  30 06 00 00 00 00 00 00 | /.......0.......
029980: 31 06 00 00 00 00 00 00  32 06 00 00 00 00 00 00 | 1.......2.......
029990: 33 06 00 00 00 00 00 00  34 06 00 00 00 00 00 00 | 3.......4.......
0299A0: 35 06 00 00 00 00 00 00  36 06 00 00 00 00 00 00 | 5.......6.......
0299B0: 37 06 00 00 00 00 00 00  00 06 00 00 00 00 00 00 | 7................
0299C0: 39 06 00 00 00 00 00 00  3A 06 00 00 00 00 00 00 | 9.......:.......
0299D0: 3B 06 00 00 00 00 00 00  3C 06 00 00 00 00 00 00 | ;.......<.......
0299E0: 3D 06 00 00 00 00 00 00  3E 06 00 00 00 00 00 00 | =.......>.......
0299F0: 3F 06 00 00 00 00 00 00  40 06 00 00 00 00 00 00 | ?.......@.......
```

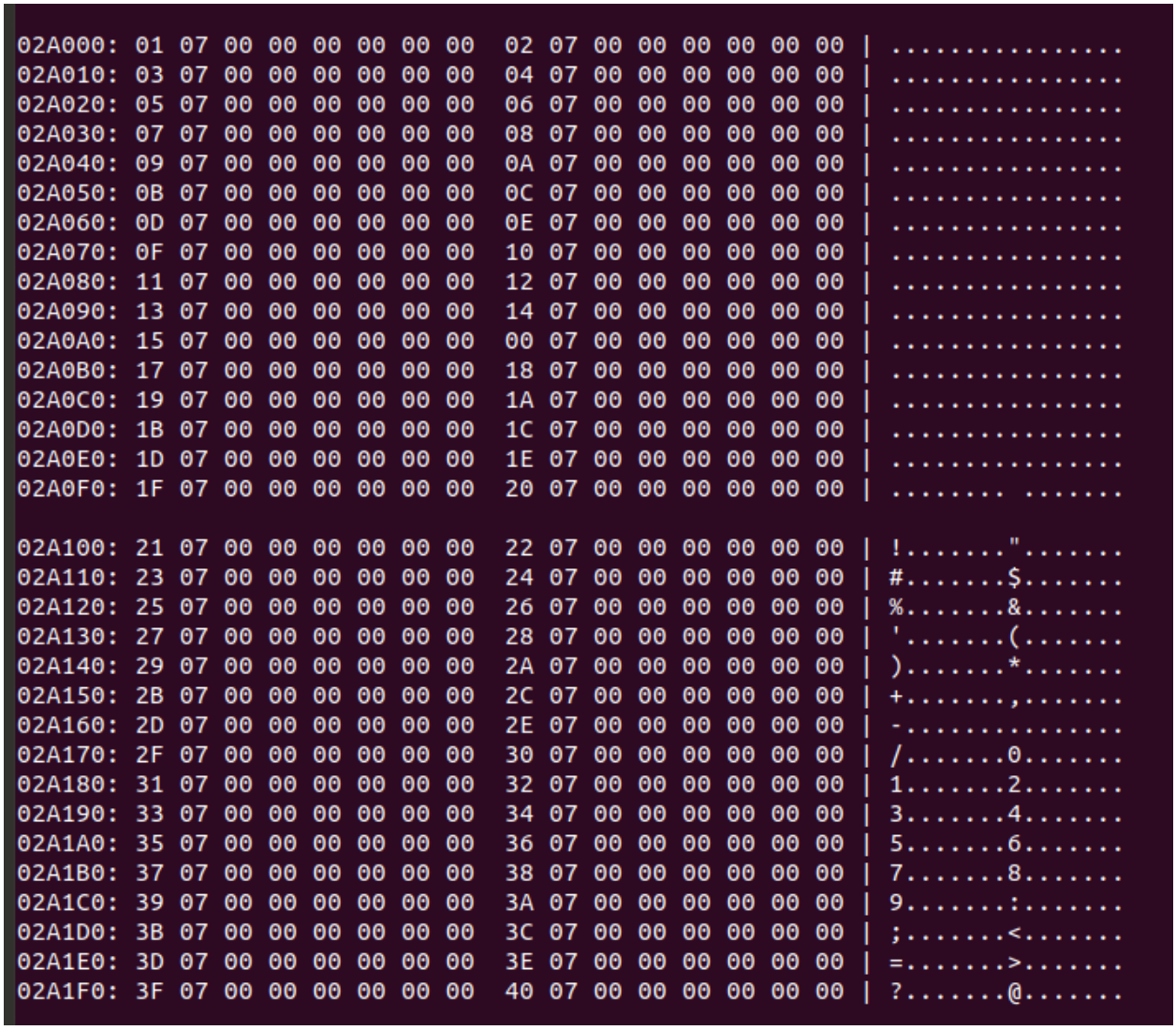


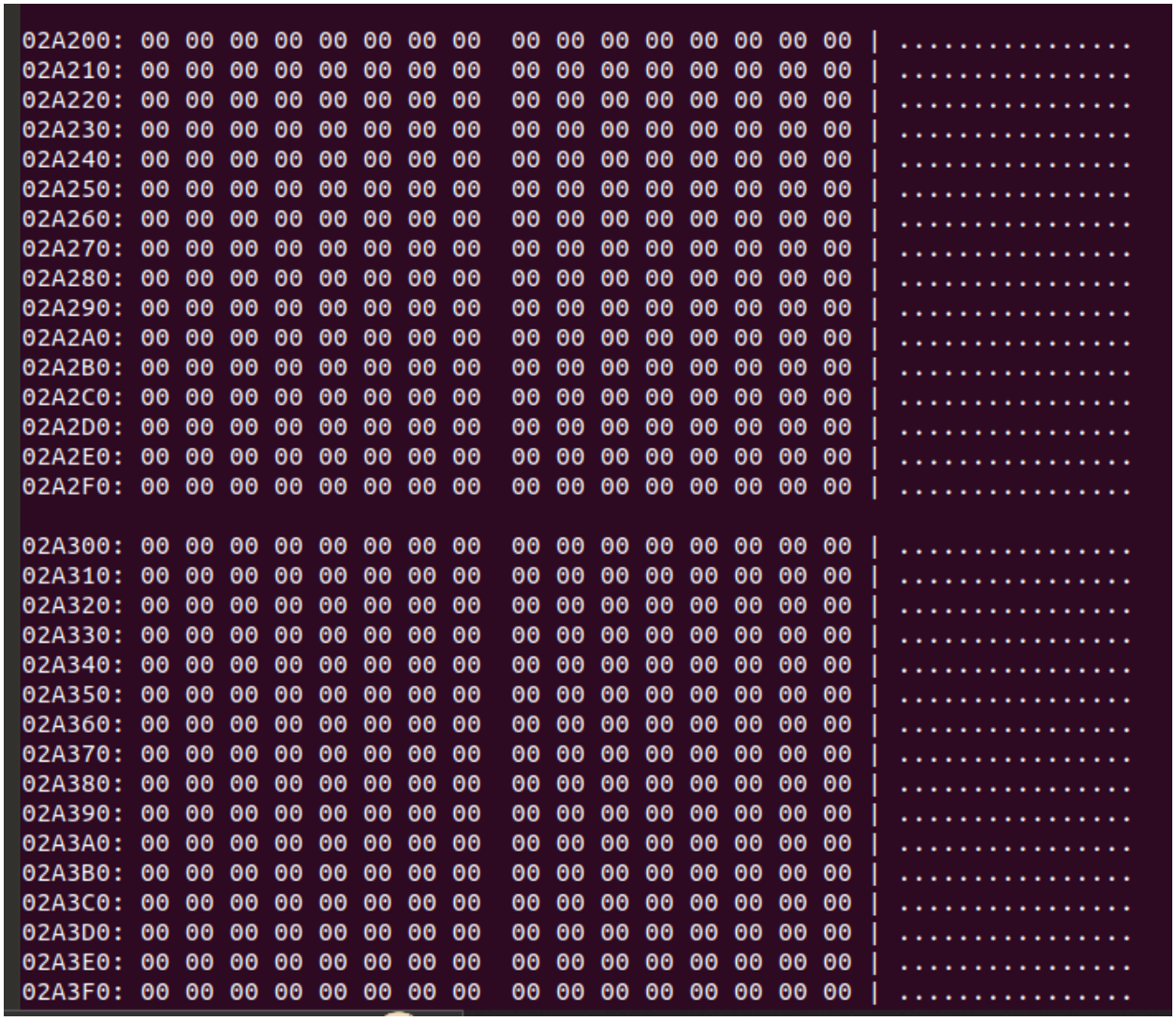
```
029A00: 41 06 00 00 00 00 00 00  42 06 00 00 00 00 00 00 | A.......B.......
029A10: 43 06 00 00 00 00 00 00  44 06 00 00 00 00 00 00 | C.......D.......
029A20: 45 06 00 00 00 00 00 00  46 06 00 00 00 00 00 00 | E.......F.......
029A30: 47 06 00 00 00 00 00 00  48 06 00 00 00 00 00 00 | G.......H.......
029A40: 49 06 00 00 00 00 00 00  4A 06 00 00 00 00 00 00 | I.......J.......
029A50: 4B 06 00 00 00 00 00 00  4C 06 00 00 00 00 00 00 | K.......L.......
029A60: 4D 06 00 00 00 00 00 00  4E 06 00 00 00 00 00 00 | M.......N.......
029A70: 4F 06 00 00 00 00 00 00  50 06 00 00 00 00 00 00 | O.......P.......
029A80: 51 06 00 00 00 00 00 00  52 06 00 00 00 00 00 00 | Q.......R.......
029A90: 53 06 00 00 00 00 00 00  54 06 00 00 00 00 00 00 | S.......T.......
029AA0: 55 06 00 00 00 00 00 00  56 06 00 00 00 00 00 00 | U.......V.......
029AB0: 57 06 00 00 00 00 00 00  58 06 00 00 00 00 00 00 | W.......X.......
029AC0: 59 06 00 00 00 00 00 00  5A 06 00 00 00 00 00 00 | Y.......Z.......
029AD0: 5B 06 00 00 00 00 00 00  5C 06 00 00 00 00 00 00 | [.......\.......
029AE0: 00 06 00 00 00 00 00 00  5E 06 00 00 00 00 00 00 | ........^.......
029AF0: 5F 06 00 00 00 00 00 00  60 06 00 00 00 00 00 00 | _.......`.......

029B00: 61 06 00 00 00 00 00 00  62 06 00 00 00 00 00 00 | a.......b.......
029B10: 63 06 00 00 00 00 00 00  64 06 00 00 00 00 00 00 | c.......d.......
029B20: 65 06 00 00 00 00 00 00  66 06 00 00 00 00 00 00 | e.......f.......
029B30: 67 06 00 00 00 00 00 00  68 06 00 00 00 00 00 00 | g.......h.......
029B40: 69 06 00 00 00 00 00 00  6A 06 00 00 00 00 00 00 | i.......j.......
029B50: 6B 06 00 00 00 00 00 00  6C 06 00 00 00 00 00 00 | k.......l.......
029B60: 6D 06 00 00 00 00 00 00  6E 06 00 00 00 00 00 00 | m.......n.......
029B70: 6F 06 00 00 00 00 00 00  70 06 00 00 00 00 00 00 | o.......p.......
029B80: 71 06 00 00 00 00 00 00  72 06 00 00 00 00 00 00 | q.......r.......
029B90: 73 06 00 00 00 00 00 00  74 06 00 00 00 00 00 00 | s.......t.......
029BA0: 75 06 00 00 00 00 00 00  76 06 00 00 00 00 00 00 | u.......v.......
029BB0: 77 06 00 00 00 00 00 00  78 06 00 00 00 00 00 00 | w.......x.......
029BC0: 79 06 00 00 00 00 00 00  7A 06 00 00 00 00 00 00 | y.......z.......
029BD0: 7B 06 00 00 00 00 00 00  7C 06 00 00 00 00 00 00 | {.......|.......
029BE0: 7D 06 00 00 00 00 00 00  7E 06 00 00 00 00 00 00 | }.......~.......
029BF0: 7F 06 00 00 00 00 00 00  80 06 00 00 00 00 00 00 | ................
```



```
029C00: 81 06 00 00 00 00 00 00  00 06 00 00 00 00 00 00 | ....????????????
029C10: 83 06 00 00 00 00 00 00  84 06 00 00 00 00 00 00 | ....????????????
029C20: 85 06 00 00 00 00 00 00  86 06 00 00 00 00 00 00 | ....????????????
029C30: 87 06 00 00 00 00 00 00  88 06 00 00 00 00 00 00 | ....????????????
029C40: 89 06 00 00 00 00 00 00  8A 06 00 00 00 00 00 00 | ....????????????
029C50: 8B 06 00 00 00 00 00 00  8C 06 00 00 00 00 00 00 | ....????????????
029C60: 8D 06 00 00 00 00 00 00  8E 06 00 00 00 00 00 00 | ....????????????
029C70: 8F 06 00 00 00 00 00 00  90 06 00 00 00 00 00 00 | ....????????????
029C80: 91 06 00 00 00 00 00 00  92 06 00 00 00 00 00 00 | ....????????????
029C90: 93 06 00 00 00 00 00 00  94 06 00 00 00 00 00 00 | ....????????????
029CA0: 95 06 00 00 00 00 00 00  96 06 00 00 00 00 00 00 | ....????????????
029CB0: 97 06 00 00 00 00 00 00  98 06 00 00 00 00 00 00 | ....????????????
029CC0: 99 06 00 00 00 00 00 00  9A 06 00 00 00 00 00 00 | ....????????????
029CD0: 9B 06 00 00 00 00 00 00  9C 06 00 00 00 00 00 00 | ....????????????
029CE0: 9D 06 00 00 00 00 00 00  9E 06 00 00 00 00 00 00 | ....????????????
029CF0: 9F 06 00 00 00 00 00 00  A0 06 00 00 00 00 00 00 | ....????????????

029D00: A1 06 00 00 00 00 00 00  A2 06 00 00 00 00 00 00 | ....????????????
029D10: A3 06 00 00 00 00 00 00  A4 06 00 00 00 00 00 00 | ....????????????
029D20: A5 06 00 00 00 00 00 00  A6 06 00 00 00 00 00 00 | ....????????????
029D30: 00 06 00 00 00 00 00 00  A8 06 00 00 00 00 00 00 | ........????????
029D40: A9 06 00 00 00 00 00 00  AA 06 00 00 00 00 00 00 | ....????????????
029D50: AB 06 00 00 00 00 00 00  AC 06 00 00 00 00 00 00 | ....????????????
029D60: AD 06 00 00 00 00 00 00  AE 06 00 00 00 00 00 00 | ....????????????
029D70: AF 06 00 00 00 00 00 00  B0 06 00 00 00 00 00 00 | ....????????????
029D80: B1 06 00 00 00 00 00 00  B2 06 00 00 00 00 00 00 | ....????????????
029D90: B3 06 00 00 00 00 00 00  B4 06 00 00 00 00 00 00 | ....????????????
029DA0: B5 06 00 00 00 00 00 00  B6 06 00 00 00 00 00 00 | ....????????????
029DB0: B7 06 00 00 00 00 00 00  B8 06 00 00 00 00 00 00 | ....????????????
029DC0: B9 06 00 00 00 00 00 00  BA 06 00 00 00 00 00 00 | ....????????????
029DD0: BB 06 00 00 00 00 00 00  BC 06 00 00 00 00 00 00 | ....????????????
029DE0: BD 06 00 00 00 00 00 00  BE 06 00 00 00 00 00 00 | ....????????????
029DF0: BF 06 00 00 00 00 00 00  C0 06 00 00 00 00 00 00 | ....????????????
```


029E00: C1 06 00 00 00 00 00 00 C2 06 00 00 00 00 00 00
029E10: C3 06 00 00 00 00 00 00 C4 06 00 00 00 00 00 00
029E20: C5 06 00 00 00 00 00 00 C6 06 00 00 00 00 00 00
029E30: C7 06 00 00 00 00 00 00 C8 06 00 00 00 00 00 00
029E40: C9 06 00 00 00 00 00 00 CA 06 00 00 00 00 00 00
029E50: CB 06 00 00 00 00 00 00 00 06 00 00 00 00 00 00
029E60: CD 06 00 00 00 00 00 00 CE 06 00 00 00 00 00 00
029E70: CF 06 00 00 00 00 00 00 D0 06 00 00 00 00 00 00
029E80: D1 06 00 00 00 00 00 00 D2 06 00 00 00 00 00 00
029E90: D3 06 00 00 00 00 00 00 D4 06 00 00 00 00 00 00
029EA0: D5 06 00 00 00 00 00 00 D6 06 00 00 00 00 00 00
029EB0: D7 06 00 00 00 00 00 00 D8 06 00 00 00 00 00 00
029EC0: D9 06 00 00 00 00 00 00 DA 06 00 00 00 00 00 00
029ED0: DB 06 00 00 00 00 00 00 DC 06 00 00 00 00 00 00
029EE0: DD 06 00 00 00 00 00 00 DE 06 00 00 00 00 00 00
029EF0: DF 06 00 00 00 00 00 00 E0 06 00 00 00 00 00 00
029F00: E1 06 00 00 00 00 00 00 E2 06 00 00 00 00 00 00
029F10: E3 06 00 00 00 00 00 00 E4 06 00 00 00 00 00 00
029F20: E5 06 00 00 00 00 00 00 E6 06 00 00 00 00 00 00
029F30: E7 06 00 00 00 00 00 00 E8 06 00 00 00 00 00 00
029F40: E9 06 00 00 00 00 00 00 EA 06 00 00 00 00 00 00
029F50: EB 06 00 00 00 00 00 00 EC 06 00 00 00 00 00 00
029F60: ED 06 00 00 00 00 00 00 EE 06 00 00 00 00 00 00
029F70: EF 06 00 00 00 00 00 00 F0 06 00 00 00 00 00 00
029F80: 00 06 00 00 00 00 00 00 F2 06 00 00 00 00 00 00
029F90: F3 06 00 00 00 00 00 00 F4 06 00 00 00 00 00 00
029FA0: F5 06 00 00 00 00 00 00 F6 06 00 00 00 00 00 00
029FB0: F7 06 00 00 00 00 00 00 F8 06 00 00 00 00 00 00
029FC0: F9 06 00 00 00 00 00 00 FA 06 00 00 00 00 00 00
029FD0: FB 06 00 00 00 00 00 00 FC 06 00 00 00 00 00 00
029FE0: FD 06 00 00 00 00 00 00 FE 06 00 00 00 00 00 00
029FF0: FF 06 00 00 00 00 00 00 00 07 00 00 00 00 00 00


02A000: 01 07 00 00 00 00 00 00 02 07 00 00 00 00 00 00 | ....
02A010: 03 07 00 00 00 00 00 00 04 07 00 00 00 00 00 00 | ....
02A020: 05 07 00 00 00 00 00 00 06 07 00 00 00 00 00 00 | ....
02A030: 07 07 00 00 00 00 00 00 08 07 00 00 00 00 00 00 | ....
02A040: 09 07 00 00 00 00 00 00 0A 07 00 00 00 00 00 00 | ....
02A050: 0B 07 00 00 00 00 00 00 0C 07 00 00 00 00 00 00 | ....
02A060: 0D 07 00 00 00 00 00 00 0E 07 00 00 00 00 00 00 | ....
02A070: 0F 07 00 00 00 00 00 00 10 07 00 00 00 00 00 00 | ....
02A080: 11 07 00 00 00 00 00 00 12 07 00 00 00 00 00 00 | ....
02A090: 13 07 00 00 00 00 00 00 14 07 00 00 00 00 00 00 | ....
02A0A0: 15 07 00 00 00 00 00 00 00 07 00 00 00 00 00 00 | ....
02A0B0: 17 07 00 00 00 00 00 00 18 07 00 00 00 00 00 00 | ....
02A0C0: 19 07 00 00 00 00 00 00 1A 07 00 00 00 00 00 00 | ....
02A0D0: 1B 07 00 00 00 00 00 00 1C 07 00 00 00 00 00 00 | ....
02A0E0: 1D 07 00 00 00 00 00 00 1E 07 00 00 00 00 00 00 | ....
02A0F0: 1F 07 00 00 00 00 00 00 20 07 00 00 00 00 00 00 | ....
02A100: 21 07 00 00 00 00 00 00 22 07 00 00 00 00 00 00 | !.......".......
02A110: 23 07 00 00 00 00 00 00 24 07 00 00 00 00 00 00 | #.......$.......
02A120: 25 07 00 00 00 00 00 00 26 07 00 00 00 00 00 00 | %.......&.......
02A130: 27 07 00 00 00 00 00 00 28 07 00 00 00 00 00 00 | '.......(.......
02A140: 29 07 00 00 00 00 00 00 2A 07 00 00 00 00 00 00 |).......*.......
02A150: 2B 07 00 00 00 00 00 00 2C 07 00 00 00 00 00 00 | +.......,.......
02A160: 2D 07 00 00 00 00 00 00 2E 07 00 00 00 00 00 00 | -................
02A170: 2F 07 00 00 00 00 00 00 30 07 00 00 00 00 00 00 | /.......0.......
02A180: 31 07 00 00 00 00 00 00 32 07 00 00 00 00 00 00 | 1.......2.......
02A190: 33 07 00 00 00 00 00 00 34 07 00 00 00 00 00 00 | 3.......4.......
02A1A0: 35 07 00 00 00 00 00 00 36 07 00 00 00 00 00 00 | 5.......6.......
02A1B0: 37 07 00 00 00 00 00 00 38 07 00 00 00 00 00 00 | 7.......8.......
02A1C0: 39 07 00 00 00 00 00 00 3A 07 00 00 00 00 00 00 | 9.......:.......
02A1D0: 3B 07 00 00 00 00 00 00 3C 07 00 00 00 00 00 00 | ;.......<.......
02A1E0: 3D 07 00 00 00 00 00 00 3E 07 00 00 00 00 00 00 | =.......>.......
02A1F0: 3F 07 00 00 00 00 00 00 40 07 00 00 00 00 00 00 | ?.......@.......


02A200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A2F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A300: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A380: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A390: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....
02A3F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ....

```
02A400: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A410: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A420: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A430: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A440: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A450: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A460: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A470: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A480: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A490: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A4F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

02A500: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A510: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A520: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A530: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A540: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A550: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A560: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A570: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A580: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A590: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
02A5F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
student@student-VirtualBox:~/Documents/Group-File-System/csc415-filesystem-westleyc30$
```

- **Hex addresses interpreted:**
    - 2E (026800): Represents “.” in ASCII, which is the root directory
    - 2E2E (026920): Represents “..” in ASCII, which is the parent directory
    - In addition, after count 336, each hex dump would become empty and just 0s, meaning we are in the leftover remaining free space in our volume

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Milestone 2

Descriptions:

1. Fs_setcwd:
    a. fs_setcwd takes in a pathname and first, checks if the pathname is NULL or if the pathname is root and if either of those are true we set the current working directory to the root directory. Then we check if the first character in pathname is "/" if it is then the path is an absolute path. If the path is relative then: first check if the current path is the root, allocate memory to the new path. We then clean the path and free all the allocated memory.

2. fs_getcwd:
    a. fs_getwd takes in a pathname, and would be setting up the current path that we're in

3. fs_isFile and fs_isDir:
    a. Both fs_isFile and fs_isDir help to act as a checker for any directory entry in our file system, detecting whether or not that given entry is a file or a directory. These two functions would be used as helper functions throughout our mfs.c, and work along with other helper functions.

4. fs_mkdir:
    a. fs_mkdir is used to help create a new directory, which means that we need to find a directory entry in the parent's array of DEs. This is used in the file system function, md.

5. fs_opendir:
    a. fs_opendir returns a file descriptor directory structure, and inputs information into that structure needed for a directory. It would also store the path information in which that

directory is. In addition, fs_opendir will load that directory into memory. This is used in the file system function, ls.

6. fs_readdir:
    a. fs_readdir acts as the ls function for a file system. It takes in the information that we made from fs_opendir, and it would try to search for any existing directory entries we made in a specific path. From there, we would get an output on the console on what directory entries we have currently in our file system. This is used in the file system function, ls.

7. fs_closedir:
    a. fs_closedir will set each of the values within the fdDir structure to null, freeing any resources we set up from fs_opendir. This is used in the file system function, ls.

8. fs_stat:
    a. fs_stat is used to help create and return certain attributes about a given directory that were stored in a structure, such as time of creation or modification, name, how many blocks it takes up, and so on. This is used in the file system function, ls.

9. fs_delete:
    a. fs_deletd would simply locate each of the given blocks for the requested file, replace each of them with some kind of character to indicate that they're all free to use. This is used in the file system function, rm, specifically when we call fs_isFile.

10. fs_rmdir:
    a. Similar to fs_delete, fs_rmdir would just replace each of the given blocks with some kind of character to indicate that they're free to use. This is used in the file system function, rm, specifically when we call fs_isDir.

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Roles:

Main functions:

Section	Who worked on it
fs_setcwd	**Westley, Natalie**
fs_getcwd	**Angelo**
fs_isFile	**Westley**
fs_isDir	**Precious**
fs_mkdir	**Natalie**
fs_opendir	**Angelo**
fs_readdir	**Natalie**
fs_closedir	**Natalie**
fs_stat	**Natalie, Angelo**
fs_delete	**Precious**
fs_rmdir	**Natalie**

Helper functions:

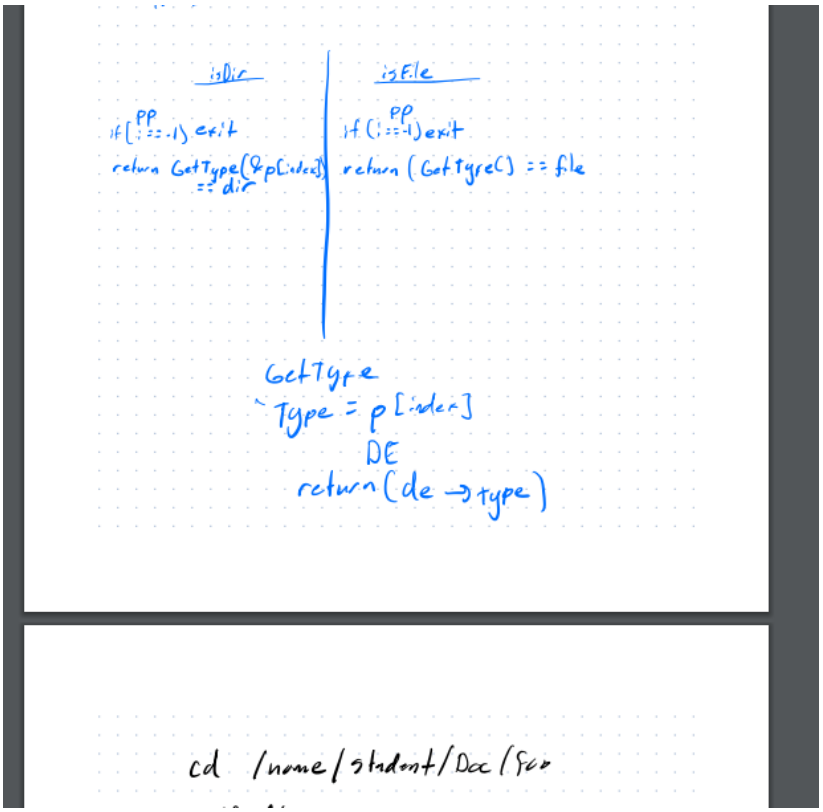
Section	Who worked on it

parsePath	**Everyone**
loadDir	**Bryan**
writeDir	**Bryan**
getType	**Natalie**
isDir(DE*dir)	**Natalie**

Approach / What we Did:

1. For the first step, just like with the last milestone, we focused on setting up our roles for each of the given directory functions. Since we have a group of 5, and there are 9 necessary functions, we decided to have least __ functions given to each group member. If helper or optional functions are needed, we would also divide up that work for each of the members, mainly giving one each in order to help with each of the other main functions.

a. In addition, we would continue to take note of any pseudo code that was given in class, such as the helper functions like GetType, isDir/isFile, and more:



b. In addition, we would create more additional helper functions that were not mentioned in any of the given lectures. Some of these include setting the root directory and current directory because once we start up the file system, we need the root and current directory to track both the . and ..

```
/**
 * Two helper functions that help set both of the root
 * and current directory, both of which are needed
 * durring the intial start up of the file system,
 * mainly for the . and ..
*/
void setRootDirParse(directoryEntry* testRoot){
    rootDirParse = testRoot;
}

void setCurrDirParse(directoryEntry* testDir) {
    currDirParse = testDir;
}
```

2. For the next step, our groups would still try to meet up as usual. Many of our meetups would often focus on debugging certain issues, reengineering some of our existing code such as the code for our directories, or we would focus on having to plan out what we need to do next.
    c. We would also have time stamps for any given lectures that might be important for us to look back on, just in case we need a reminder for certain functions

File System Lectures Time Stamps

Oct 9 - milestone 1 introduced 37:42

Oct 11 - goes over the file system git hub 17:58

Oct 16 - goes over initRootDir() and shows his freeSpace.h file

Oct 18 - Starts paging and stuff

Oct 23 - continues paging

Oct 25 - goes over assignment 5 -> at 41:28 starts talking about milestone 2
- mfs.h is the main focus of this milestone -> directory functions
- getcwd() -> when you first start out the directory you are in is root, we use strncopy to copy the name from fs_diriteminfo
- fsdelete() -> We have to free the blocks associated with the file then mark that entry as unused
- Fsstat struct has nothing to do with our directory entry, it is a return structure for the function fsStat()
- Mkdir()

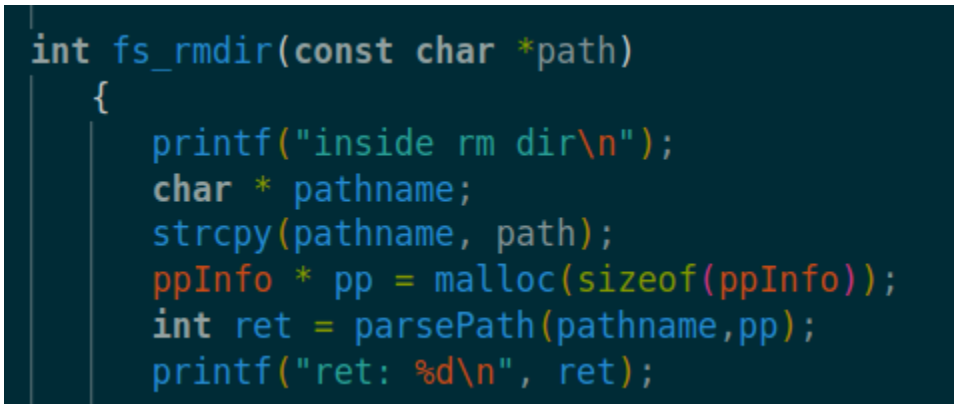
Oct 30th - ParsePath

Nov 1 - more paging stuff

Nov 6 - mkdir() again, goes over milestone 3: file operations, rmdir() vs fsDelete @ 39:00min, more milestone 2 functions: 51:45min (clean path, ls etc..)

Nov 8 - CPU scheduling

3. While debugging each of the functions in milestone 2, our test runs would mainly involve using functions from fsshell.c. One example comes from mkdir and rmdir, in which we needed to turn on both the md and rm functions in the file system to see if we actually created a directory and loaded it to disk, and if it was actually removed. At the same time, we would also implement many print statements, most of which us indicate whether or not we're traversing through that function, if the argument for that function is passing through separate helper functions, or if it even reaches to the end of the given function



```
int fs_rmdir(const char *path)
    {
        printf("inside rm dir\n");
        char * pathname;
        strcpy(pathname, path);
        ppInfo * pp = malloc(sizeof(ppInfo));
        int ret = parsePath(pathname,pp);
        printf("ret: %d\n", ret);
```

- In addition, we would also create various test dummies for some of our functions within milestone 2, such as making a dummy path and we would call fs_mkdir and parsepath that would help us see whether or not the path was created starting from root. Doing this allowed us to see whether or not mk_dir or parsepath would work or not. This, and many other milestone 2

functions would have similar testing scenarios that utilized dummy paths and directories.

```
// Testing here
// directoryEntry* test = loadDirectory(rootDir);
// rootDir = NULL;
// printRootDir();
// printCurrDir();
//printf("result from getEOF on root is :%d\n", getEOF(rootDir) );

// const char* testPath = "/home";
// char* testPathParse = "/home";
// int result = fs_mkdir(testPath, 0);
// ppInfo* testInfo = malloc(sizeof(testInfo));
// int ret = parsePath(testPathParse, testInfo);
// printf("ret is:%d\n", ret);

// directoryEntry* homeDir = &(testInfo-> parent[testInfo->index]);
// printf("homeDir isDir is :%d\n", homeDir->isDir);
// int homeEOF = getEOF(homeDir);
// printf("homeEOF before append is:%d\n", homeEOF);
// int appendRet = appendBlocks(homeDir, 50);
// printf("appendRet:%d\n", appendRet);
// homeEOF = getEOF(homeDir);
// printf("homeEOF after appendding 50 blocks is:%d\n", homeEOF);

// Test ends
```

4. For the final step, once we managed to debug a majority of the functions for mfs.c, including various helper functions, we would then clean up a lot of leftover test code that was commented out, printf statements, add a bit more documentation, format the layouts of each file and put in the proper headers.

Approaches for Individual functions for Milestone 2

- fs_getcwd and fs_opendir:
    1. For both of these given functions, since some of the pseudo code was given through class, we used it as a template to figure out how to implement the functionality of both functions.
- fs_stat:
    1. How we managed to come up with fs_stat was by reading some of the comments made for the given structure that we're supposed to return. By reading it, it helped to understand that the only thing we're doing is just giving values to each of the members in the stat structure, while just doing a parsepath.
    2. Doing this, we managed to simplify the execution and layout of code needed to create fs_stat.

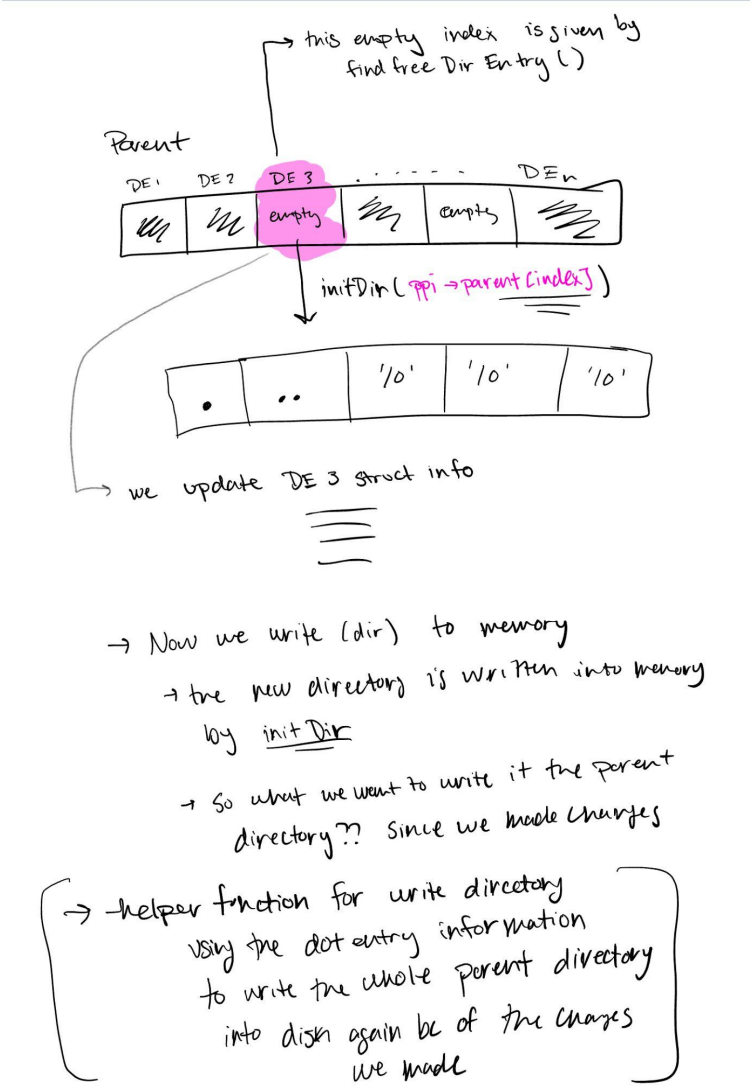
- Fs_setcwd:
    a. For setcwd we knew that we would need to update the global variables that were keeping track of what DE we were currently in both in string form(the pathname) and the actual directory entry. For us that was the cwd and curDirParse variables. We knew that we would need to call parse path on the desired path that was given by the user. In order to do this we first need to create a new path name to feed into parse path, and this new path would be the concatenation of the current working directory and the given path. Once we pass this newpath into parse path we can use the parent field in the ppi to set our curDirParse global variable. Now we need a way to update our cwd variable which is used in the pwd command. This should be a clean version of the newpath that is "nice looking". In order to do this we need a helper function, here is a picture of how we thought about implementing this cleanPath function:

```
CleanPath (char *path) {

    char * savePtr = NULL;
    char * token = (path, "/", &savePtr)

    if (token == NULL)
        if (strcmp (path, "/") == 0){
            return path
        }

    char ** tokenArray = malloc( )

    while (token != NULL) {
        if (strcmp (token, ".") != 0) {
            tokenArray[i] = token
```

- fs_rmdir:
    1. For rmdir we knew that we needed to mark the blocks that the directory was occupying as free again. To do this we first needed to check whether the directory given to remove actually existed. We used ppi from parse path to check whether the given directory exists and from there we used the helper function freeBlocks() and passed in the directory entry to free the blocks it was using. Most of the work for mk_dir was done by our helper functions ParsePath and freeBlocks.
- fs_isFile:
    1. The isFile function was relatively simple to approach given that our DE struct had a field that marked whether or not the DE was a directory. Using this in combination with

parsepath allowed us to attain the directory entry and check its field to return whether it was a file or not.

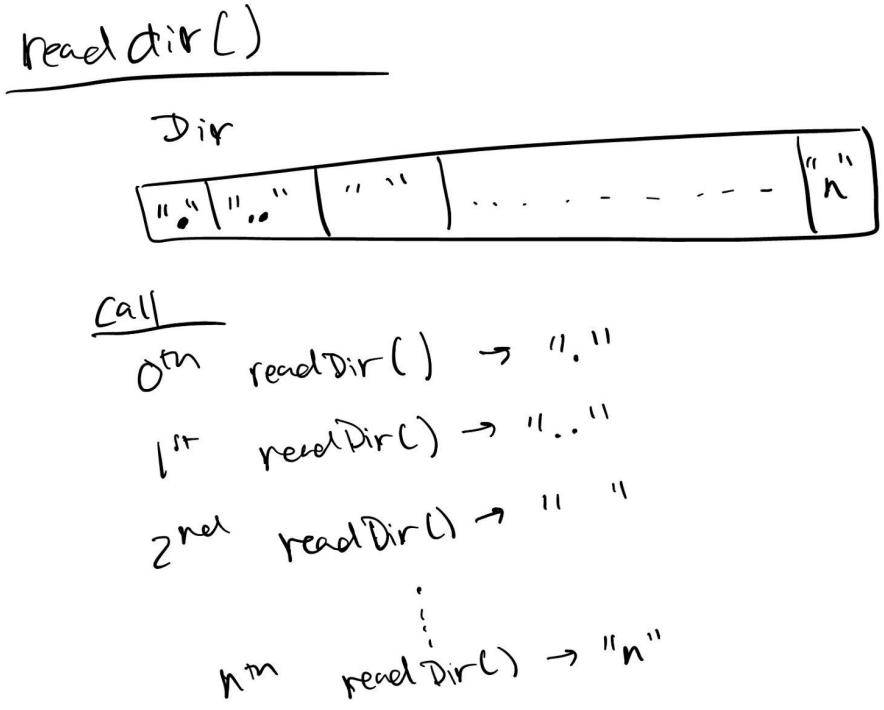
- fs_isDir:
    1. The isDir functions works almost exactly the same as the isFile function except in this case we return whether the given DE is a directory.
- fs_mkdir:
    1. Below is a drawing for our approach for implementing the mkdir function and deciding what helper functions we needed in order to do this:



From drawing out our thought process we realized that we would need two helper functions, one for finding a free directory entry in the parent array and one for writing the parent directory over again into disk with the new directory entry struct fields filled in with what the user wanted.

- fs_readdir:

1. We started tackling the readdir function by first trying to figure out what or who was going to be using read dir to get a better understanding of what its functionality was supposed to be. We discovered that the read dir function was used in displayfiles() in fsshell.c. The function of readdir it to iteratively read a DE in the parent and get the names so that ls can display them. Once we knew this, we knew we would need a for loop to iterate through the parent's DEs to find the nonempty ones and set the directory entry position field in the fddir struct to the correct index in the parent array for the next call to readdir(). Here is how we thought about iterating through the parent:



- fs_closedir:
    1. We thought of closedir as the compliment of opendir, meaning that the fdDir struct we initialized in opendir should be cleaned up by freeing whatever was necessary and nulling any pointers as well.

- fs_stat:

1. Our thought process behind implementing this function was to first find out how it was going to be used in fshell to gain a better understanding to its functionality. We discovered that it was used for one of the ls helper functions, displayFiles(). Once we understood this we knew the functionality of fsstat was to fill in the stat buf with the necessary details from the DE of the directory we were calling parse path on, specifically once we knew this struct was going to be used for ls -l in particular the fs_stat() function made a lot more sense. From there we knew that we needed to call parse path and get the DE that we were going to use to fill in the information that ls -l needed to display the correct contents.

- fs_delete:
    1. This function was almost the same as fs_rmDir, so the thought process was similar. The only difference here was checking the DE for the given path ending element was a file instead of a directory. From there the steps were the same, free the blocks associated with the file using freeBlocks(), update the name of the DE for the file to null to indicate the DE could be used for something else now and then call writeDir to write the new changes we made into disk.

- fs_rmdir:
    1. The thought process for removedir was very similar to mkdir, we needed to use parsepath to see if the the directory asked to be removed existed. Once we are sure of that then we can proceed to use our helper functions to clear the blocks that the DE was occupying we use our helper functions isDir to make sure what we are trying to clear actually is a directory, freeBlocks() to mark the blocks as free to use again and finally writeDir to write the new changes to the parent directory back to disk.

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Issues and Resolutions (Debugging related issues):

- **Issue #1:** The first issue we came across was getting a conflicting type error when passing in our header file for our directories into mfs.h. We needed to do this because one of the structs, mainly fdDir, needed a pointer to the directory entry that we're reading from.

```
In file included from mfs.h:22:0,
                 from fsshell.c:29:
directories.h:36:3: error: conflicting types for ‘directoryEntry’
 } directoryEntry; // size is 296 bytes
   ^~~~~~~~~~~~~~
In file included from mfs.h:20:0,
                 from fsshell.c:29:
directories.h:36:3: note: previous declaration of ‘directoryEntry’ was here
 } directoryEntry; // size is 296 bytes
   ^~~~~~~~~~~~~~
In file included from mfs.h:22:0,
                 from fsshell.c:29:
directories.h:38:5: error: conflicting types for ‘initDir’
 int initDir(directoryEntry * parent);
     ^~~~~~~
In file included from mfs.h:20:0,
                 from fsshell.c:29:
directories.h:38:5: note: previous declaration of ‘initDir’ was here
 int initDir(directoryEntry * parent);
     ^~~~~~~
```

Resolution #1: How we resolved this issue was by changing the syntax for the header file for our directories.h. Since we called this header file from various files

- **Issue #2**: Another issue that we have encountered came from our hexdump, in which our vcb was in the wrong starting block for the hexdump, as it was placed in block 2 instead of block 1 as instructed.

  Resolution #3: How we resolved this issue was by noticing that we had two mallocs for the fsInit.c function. By erasing one of them, we managed to get the VCB back into place, mainly in the hexdump that starts at 1 or virtual address 0x000200

```
/* STEP 1: initializing the VCB */
vcb_p = malloc(blockSize); // creating an instance of the vcb struct
```

- **Issue #3:** Another issue that our group came across was mainly with various warning messages, and various undefined functions that we placed in our mfs.c file. After putting our functions for mfs.c into the same file, some of these issues come from either passing incorrect arguments in certain functions, incorrect comparison values, or just incorrect arguments altogether.

```
mfs.c: In function ‘fs_opendir’:
mfs.c:206:17: warning: passing argument 1 of ‘parsePath’ discards ‘const’ qualifier from pointer target type [-Wdiscarded-qualifiers]
       parsePath(path, ppi); //Called parsePath
                 ^~~~
mfs.c:87:5: note: expected ‘char *’ but argument is of type ‘const char *’
 int parsePath(char * path, ppInfo * ppi)
     ^~~~~~~~~
mfs.c:211:20: warning: return makes pointer from integer without a cast [-Wint-conversion]
            return -1;
                   ^
mfs.c: In function ‘fs_isFile’:
mfs.c:247:21: warning: passing argument 1 of ‘isADirectory’ makes pointer from integer without a cast [-Wint-conversion]
   if (isADirectory( &(currDirParse[index]) == 0)) {
                     ^
```

```
mfs.o: In function `fs_mkdir':
/home/student/Documents/Group-File-System/csc415-filesystem-westleyc30/mfs.c:485: undefined reference to `loadDir'
mfs.o: In function `fs_readdir':
/home/student/Documents/Group-File-System/csc415-filesystem-westleyc30/mfs.c:515: undefined reference to `isUsed'
collect2: error: ld returned 1 exit status
Makefile:62: recipe for target 'fsshell' failed
make: *** [fsshell] Error 1
```

Resolution #3: This issue was mainly resolved through debugging certain parts of mfs.c, such as calling the correct functions needed for loadDirectory, or replacing certain arguments that would function correctly.

```
fdd->directory = loadDirectory(&(ppi->parent[ppi->index]));
```

```
directoryEntry* result = loadDirectory(ppi->parent);
setCurrDirParse(result);
cwd = "/";
```

- **Issue #4:** One of the most common issues that we came across when working on the milestone 2 functions was the many segmentation faults that we came across. One of them was using our LS function, which goes over 25k different total entries. When running this function however, we would come across a segmentation core dump. When setting that value to 10 however, the function would run like normal and list out 10 different indices.

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Team: Blob

CSC 415-03 Operating Systems

Milestone 3

Descriptions:

1. b_open:
    a. b_open allows us to keep track of a certain file that was created through a structure, inputs each of the necessary information for that structure, and based on what function is called in fsshell.c, b_open would initiate any of the necessary flags that other functions in b_io need such as b_read or b_write. This is used in the fs functions touch, cat, cp, cp2l, and cp2fs
2. b_seek:
    a. b_seek allows us to change the pointer in any file we created. It would take an existing file descriptor, and would shift the pointer to various points on the file, such as changing it to the beginning, to the current position it's in from any other function such as b_read, or to the end of the file.
3. b_write:
    a. b_write allows us to modify the contents of any file we create, mainly our file systems file. Doing so would take up a certain amount of blocks and it would contain certain data that is written into disk. This function works with cp2fs and cp.
4. b_read:
    a. b_read allows us to only read the contents of a file we created within our file system. This would be used with certain functions such as cat, cp2fs, and cp.
5. b_close:
    a. b_close allows us to free up any resources that are taken up by our b_io structure for the file control block, along with any additional members that are tracked with each file descriptor. This function works with touch, cat, cp,cp2l, and cp2fs.

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Roles:

Main functions:

Section	Who worked on it
fcb struct	**Angelo and Precious**
b open	**Westley and Natatlie**
b seek	**Natalie**
b write	**Natalie and Angelo and Bryan**
b read	**Natalie and Angelo**
b close	**Angelo**

Helper functions:

Section	Who worked on it
directoryEntry * handleCreateFlag	**Natalie**

Flags:

	Flags for section	Who worked on it
Flags for b open	- **create** - **truncate** - **append**	- **Angelo and Natalie (For create and truncate)**

		- **Natalie (For Append)**
Flags that have to be kept tracked of (you might need to keep them in your file control block)	- **ROnly** - **WOnly** - **RW**	- **Angelo and Natalie**

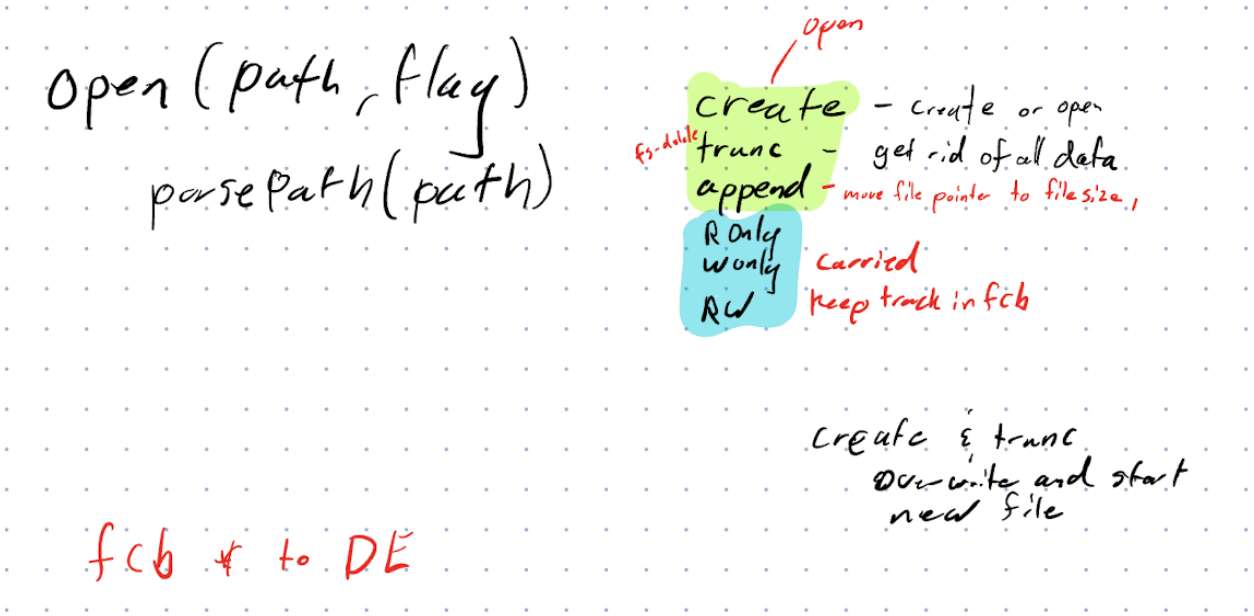
Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Team: Blob

CSC 415-03 Operating Systems

Approach / What we Did:

1. For the first step, we would still watch the given lectures that talked about milestone 3, which was November 6th. We also take any notes, such as what flags are specified for a given function, or what flags we would have to carry throughout the entire system and keep track of.



2. Once we watched the given lecture, we would later go back to our previous assignment, assignment 5, and check for any given similarities and try to reuse what existed in that assignment. A big example comes from the variables we have to keep track of in our structure, b_open, or in b_close.
3. For the next step, while having each function be given as its own role for many of our members, some of us would tackle the functions together like in milestone 2, with a sub-set of roles given such as pseudo-code, creating a foundation/layout for how our code would look, grabbing lecture material to scan through and find anything that would benefit us, and once we managed to have a good understanding of how it works, we would code it along with each other.
    a. One example comes from the helper function parsepath, as we had someone find the lecture where the pseudocode was showcased from the professor, we had someone type in

that pseudocode with slight edits that align with some of our variables, and we would have someone code over a video call while each of us would look over and declare any changes needed that would help with our functionality.

b. Along the way, our team would also try to create each of the given flags needed to help run or ignore certain b_io functions. Since some of these functions were smaller, even if some required helper functions in order to run, we would assign at most 2 people to just work on these flags.

4. For the final step, just like with milestone 2, once we managed to debug a majority of the functions for mfs.c, including various helper functions, we would then clean up a lot of leftover test code that was commented out, printf statements, add a bit more documentation, format the layouts of each file and put in the proper headers.

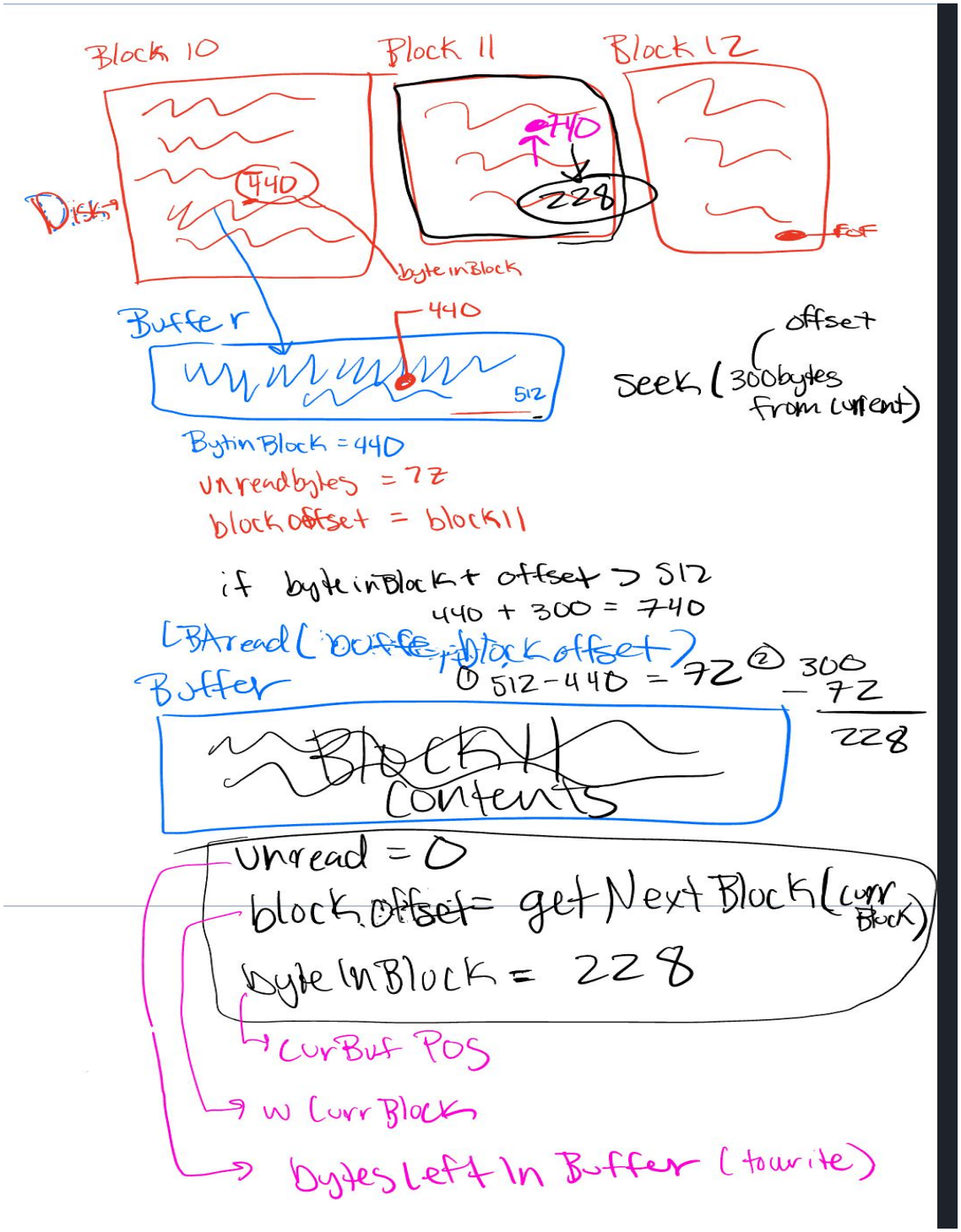
Approaches for Individual functions for Milestone 3:

- FCB_struct:
    1. How we managed to come up with the given members for our fcb_struct was by looking back on our members for assignment 5. However, since assignment 5 had its own helper function to grab information for any file, such as name, location, and size, we would do that by using our parse path.

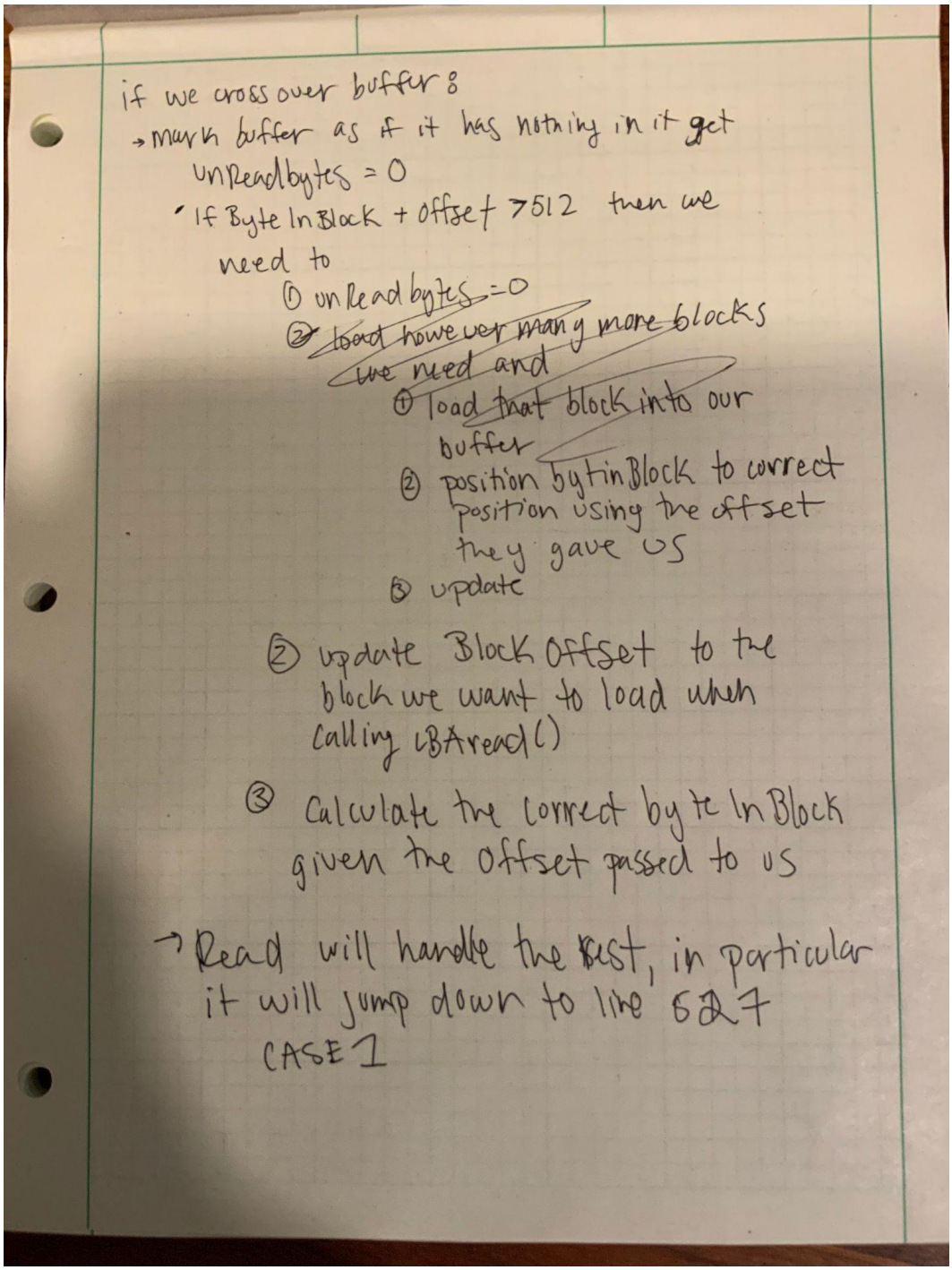
- B_Open:
    1. Just like with FCB_Struct, we would take similar elements from assignment 5 and incorporate it into the B_Open function. However, considering we also needed to implement flags, such as create, truncate, and append, that's where our approach compared to assignment 5 changes

- B_Seek:
    1. Here is a picture we drew to understand what we needed to do in each scenario of b_seek():

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan



Here is a picture of the pseudo code we came up with for b_seek() as well:



- B_Write:
    1. For this function, rather than taking the approach from one of our assignments in assignment 5, or taking the professor's approach by using parts, we decided to start from scratch. Two of our teammates would be working building the approach, simply by drawing out a visual of how our approach would work, and find scenarios for b_write which are similar to B_read. Here is more for our approach to b_write that we made during one of our meetings:

```
/**
   * CASE 1: We have nothing in our buffer
   * CASE 2: We have something in our buffer
   * CASE 3: We don't have enough space in our buffer
   * Error cases:
   *      - If the user ask for a count ask for a count that is less than 0
   *
   *      - With write, since we're only writing to the end of the file, our append function would always
   *        set the location to filesize (or just hit the last block of the file)
   *
   *      - From there, we would allocate a certain amount of blocks based on the user's count request.
   *        And from there, we would either lbawrite a certain amount of data, or just memcpy less than
   *        a block's worth straight to our buffer until we exhaust it and lba write to the user's buffer
   *
   */

  // if getEOF = currentBlock in our fcb AND the amount of then we need to allocate more blocks?
  // if (count < remainingBytesInBuffer)
  // if count > 512
  // we can only lba write at least 1 block at a time aka 512 bytes
     // wait to fill our own buffer to 512 before writing
     // scenario:
     // count = 600 bytes
     // directly write 512 or 1 block to the disk
     // allocate another block, update currentBlock field in fcb
```

```
// 88 bytes left which we would put in our buffer, user -> our buffer
// wait for more bytes from the user
// write once we fill up our buffer to 512
// uint64_t LBAwrite (void * either userBuffer if greater than or equal to 512 or OWNbuffer if less than 512, uint64_t lbaCount, uint64_t lbaPosition);
// current file position is 0 when we open
// part 1 = bytes we can write from OUR buffer to the disk DIRTY BUFFER
// part 2 = whole blocks that can be written directly from USER to the disk
// part 3 = remaining x is less than a block size CLEAN BUFFER, scenario where we have to refill buffer

// block oreintaiton to byte orientation
// we want to go from byte orientation to block orientation
When bytesLeft to write is > bytesLeft in our buffer
// rewrite that block and now it would be full
        // get the next block and set the fcb field to this block
        // handle whatever remainder there is, so we need to calculate this remainder
        // actaully we can just update BYTESWRITTEN to however much we
        // we able to fit in our buffer and then go over the loop again

        // we would then land in the case where our buffer is clean and either
        // the bytesLeft to write is < 512 or > 512.
        // if it is greater than 512 then we would need to go into a loop for
        // howver many blocks and use bryans functions to deal with writing mutliple blocks
```

- B_Close:
    1. Similar to B_Close, we would just copy over similar contents from assignment 5, such as freeing each of the blocks in our fcbArray, and having to free any buffers that were given for our fcb. However, we would also add onto freeing various pointers that we created,

Flags for Milestone 3:

- **CREATE:**
    1. We look back on the same lecture, and we take note of what happens during create, but also relate it back to what type of file system we are using.
    2. Once we watch that lecture, we would take a look into our own code, such as how we create files (mainly with the use of our directoryEntry structure), where we are supposed to put in each of our files, and how much space they would take. We'd also include some pseudo code as a way to showcase our logic based on what we made and what we need to do in order to get the create flag to work

```
// checking flags:
if(flags & O_CREAT){
    // create flag was set
    // we want to get an empty DE in the directory the user wants
    // to make the file in

    // we need to get the first free block in freespace in order to
    // allocate some blocks for the file the user wants to create
    // and then we can fill in the location field for the new files
    // DE
    // first empty block of file -> int location;
}
```

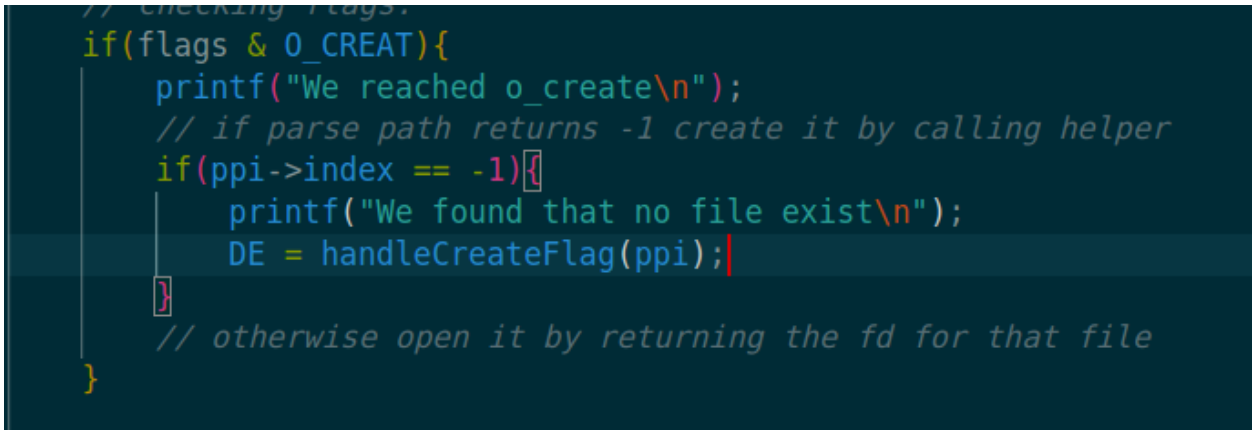
    3. Next, we would start creating our helper function that allows us to create the file if it does not exist, and that would be passed in our if statement for b_open. We would also create plain English/pseudocode for this part in order to get an idea of how we create the file in a specific directory for the user.

```
// create flag helper function:
directoryEntry * handleCreateFlag(ppInfo * ppi){
    // create flag was set
    // we want to get an empty DE in the directory the user wants
    // to make the file in

    // we need to get the first free block in freespace in order to
    // allocate some blocks for the file the user wants to create

    // file is new so the size is going to be 0

    // and then we can fill in the location field for the new files
    // DE
    // first empty block of file -> int location;
```

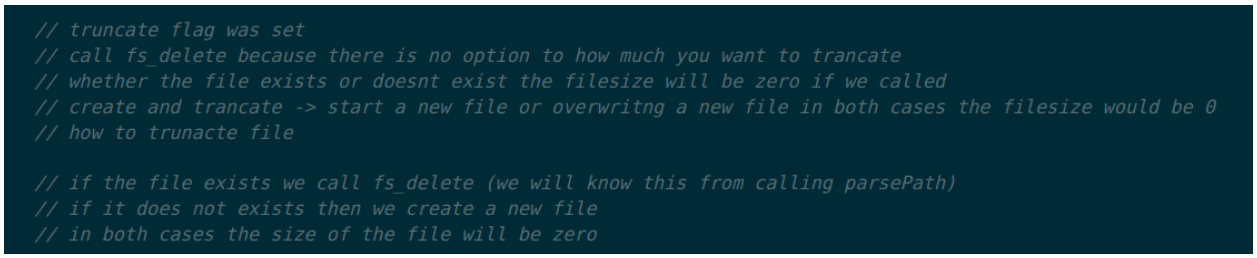
4. Moving on, once we had a good idea of what we're trying to accomplish with the helper function, we would simply follow a similar idea for b_open, since we're trying to make a new file, all we needed to do was instantiate the necessary variables that would create a file, such as being an existing entry within the given path, size, time related attributes, and so on. Once we managed to do that, we would use another helper function that would allow us to write our newly made file out onto disk.

```
// 1. find free DE in parent
directoryEntry * parent = ppi->parent;
int freeIndex = findFreeDirEntry(parent); // we get back the index of free DE
directoryEntry * newDE = &(parent[freeIndex]); //new DE for file
time_t t = time(NULL);
// fill in fields for DE:
newDE->location = allocateBlocks(1);
strncpy(newDE->name, ppi->lastElement, strlen(ppi->lastElement));
newDE->size = 512;
newDE->created = t;
newDE->modified = t;
newDE->accessed = t;
newDE->isDir = 0;

//We write the file to disk for it to be stored in our file system
writeDir(ppi->parent);
```

5. Going back to the if statement for create in our b_open function, we would simply just use this helper function just in case if the file itself does not exist within the disk



- **TRUNC:**
    1. For the truncate, we approached it by following the certain cases listed within the lecture, mainly when it came to a file existing or not existing. So we had to make sure that truncate would work in relation to the create flag. In addition, we would write plain English comments as a way to structure out our logic, just like with create



2. In addition, since a lot of elements from the create flag are similar when calling truncate, we would carry over those elements, such as finding an invalid path within our file system. And based on one of the lectures, since we would have to completely overwrite the file that truncate is calling on, we would just use one of the given functions from mfs.c, fs_delete, in order to truncate the file.

```
if(ppi->index == -1){
    // the file does not exist
    printf("file does not exist to truncate\n");
    return -1;
}else{
    //Used to free any blocks the file we're truncating took up
    int ret = fs_delete(filename);

    //Catch case if fs_delete did not work properly
    if(ret == -1){
        printf("fs delete did not work\n");
    }
}
```

We would also try to take care of a file that does not exist, and we would also use our create helper function to help take care of that, and truncate from there.

```
if ((flags & (O_TRUNC | O_CREAT)) == (O_TRUNC | O_CREAT)) {
    // Both O_TRUNC and O_CREAT are set
    // 1. check if file exists with ppi
    if(ppi->index == -1){
        // the file does not exist
        DE = handleCreateFlag(ppi);
    }else{
        int ret = fs_delete(filename);
        if(ret == -1){
            printf("fs_delete did not work\n");
        }
        DE = handleCreateFlag(ppi);
    }
}
```

- **APPEND:**
    1. For append all we needed to do was set the file pointer in our fcb to the end of file, the read and write functions would take care of the rest.

```
if(flags & O_APPEND){
    // append flag was set
    // preset the pointer to the end of the filesize aka update the fcb
    //  pointer to point to the end of the file
    fcbArray[fd].curBufPos = fcbArray[fd].DE->fileSize;
    fcbArray[fd].byteInBlock = fcbArray[fd].DE->fileSize;
}
```

Issues and Resolutions (Debugging related issues):

1. **Issue #1:** The first main issue we came across was related to the create flag in b_open. It was that we were using our ppi parent to store each of the members in open, if there was a flag set for any of them, ppi parent would not have the updated /accurate versions we set in order to fill in the file control block.

   Resolution #1: So then we would make a local variable for our DE in order to keep track of any given updates, rather than resetting them back to their original states.

```
// get our own file descriptor
// check for error - all used FCB's
directoryEntry * DE = ppi->parent;
// checking flags:
```

2. **Issue #2:** The second issue that we came across was when we were trying to create and truncate flags, as we needed to use certain helper functions in order to use the touch and rm functions. One of these helper functions, mainly fs_delete from mfs.c, we would use for our rm function in the terminal. However, we would often get a segmentation core dump.

```
        inside isADir()
Makefile:68: recipe for target 'run' failed
make: *** [run] Segmentation fault (core dumped)
student@student-VirtualBox:~/Documents/Group-File-System/csc415-filesystem-westleyc30$
```

   How we managed to resolve this issue came from replacing a certain line which belonged to our fs_isFile function, in which we tried to check if the given item did return a certain number to see if it was considered a directory or not. And that value would be passed into isADirectory and would be checked if it equaled 1 to see if it was a directory.

```
//Another catch case if the user request a directory ins
if (isADirectory(DE->isDir) == 1) {
    printf("%s is a directory not a file.\n", filename);
    return 0;
}
```

However, inside of isADirectory, we noticed that we are passing that member already, given that the argument itself is of directoryEntry, which is the file that we're checking to see if it's a directory or file.

```
int isADirectory(directoryEntry *parent)
{
    printf("\tinside isADir()\n");
    printf("Parent isDir: %d\n", parent->isDir);
    if (parent -> isDir == 1) {
        return 1;
    } else {
        printf("\tnot a directory\n");
        return 0;
    }

}
```

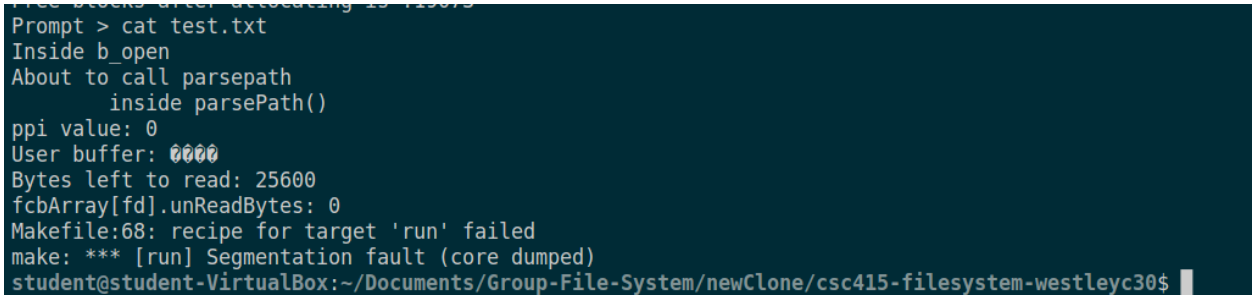
By removing that member in fs_isFile, we managed to get rid of the segmentation fault and the file was able to be removed.

3. **Issue #3:** Another issue that our group faced was that one of our members had faced a git error message that regarded certain git objects being corrupted. This issue may have appeared when one of us accidentally ended a git operation while it was still in progress, which caused the corruption.

```
student@student-VirtualBox:~/Documents/Group-File-System/csc415-filesystem-westleyc30$ git status
error: object file .git/objects/23/d5a96d5f13abe1a379fdead883b4196df408ac is empty
error: object file .git/objects/23/d5a96d5f13abe1a379fdead883b4196df408ac is empty
fatal: loose object 23d5a96d5f13abe1a379fdead883b4196df408ac (stored in .git/objects/23/d5a96d5f13abe1a379fdead883b4196df408ac) is corrupt
```

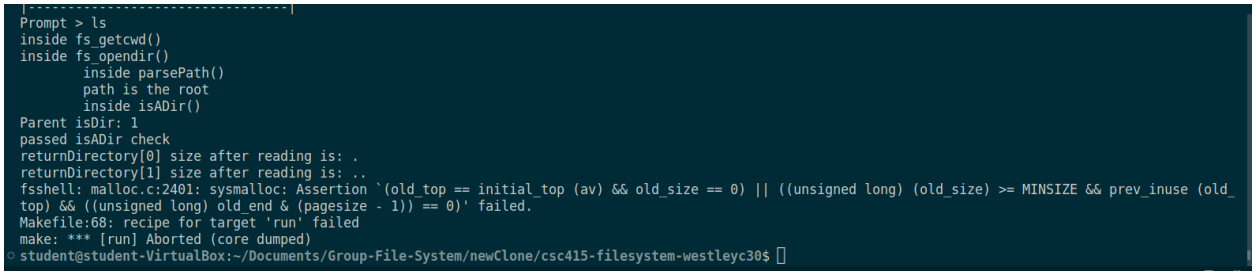
Resolution #3: While the solution itself wasn't solved appropriately, the only way we could go around it was by cloning the project since their local repository was the latest one pushed. From there, we were able to commit or push once again.

4. **Issue# 4:** Another issue that our team ran into focused on using the cat command, in which our program would have a segmentation fault, and that our filesize for the file we modified always stays at 0.



Resolution# 3: How we fixed this was by adding a field in our directoryEntry structure, mainly the fileSize member. This allowed us to save the file size into disk, because before, we would just save it in the fcb and that wouldn't save the file size between different opens and closes for a file.

5. **Issue #5:** One issue that we ran into was having our ls function cause a core dump issue. This issue occurred after getting a current working version for cp2fs and cp2l.

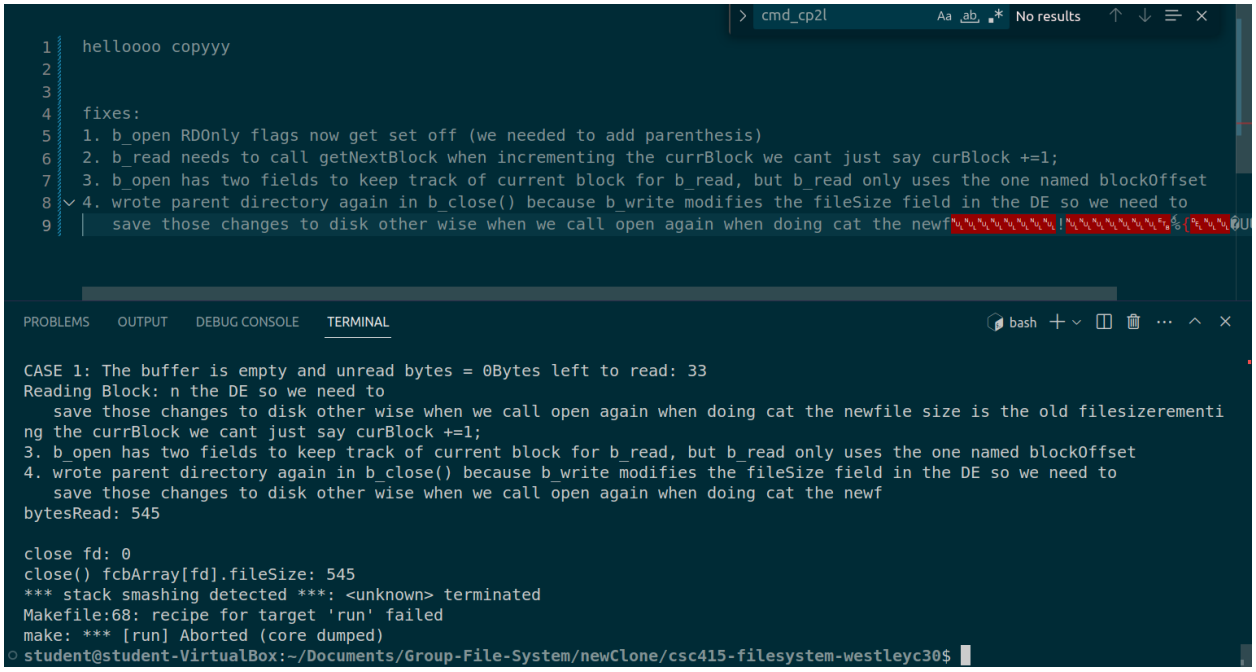


This ultimately occurred due to our directory structure having a new member that would keep track of the file size for any given directory entry.

```
38  38    typedef struct directoryEntry{
39        -   char name[472];// 472 for 512 size, 256 for 296
      39  +   char name[468];// 472 for 512 size, 256 for 296
40  40        int location;
41  41        unsigned int size; //size in bytes of directory
42  42        unsigned int fileSize; //size in bytes for the file we're using
```

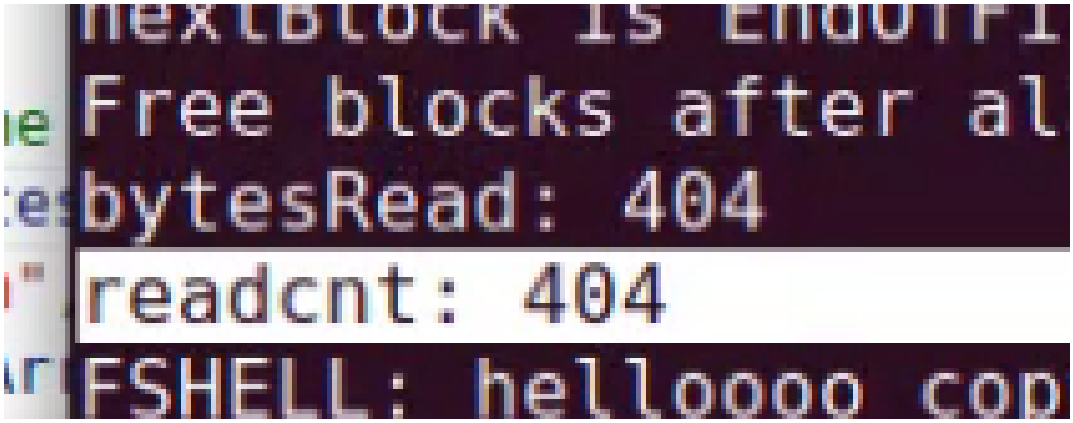
Resolution #5: How this issue was fixed was by changing how much our name member could hold, since one of our helper functions was meant to allocate blocks that are only 512 bytes. Adding our unsigned filesize member increases that byte size to 516 bytes however, causing us to go past a certain limit of memory. So by decreasing how much our name would take up to just 468 bytes, we can go back to our previous 512 bytes.

6. **Issue #6**: The next issue that we ran into was using the write function. In this issue, if we had more than a block's worth of data to write to a linux file, it would cause a null characters to appear, as well as a stacking smash detected error to occur. The stacking smash error occurs whenever use the cat command as well



With fsshell.c, we found out that the linux read function is being used for the cp2l, and it only takes in 200 bytes at a time before going to the end of the file. If we have 200 bytes, or a certain amount of bytes that would be divided up equally by 200 without any leftovers, we wouldn't be seeing any null characters in the output of the given linux file if we wrote to it with our file system file. So we assumed that it was intentionally done that way.

We also found out that stack smashing error must have come from a buffer overflow error, in which we read over a certain amount that's way more than what the user requested straight into the user's buffer. This mainly occurs in the b_read function. So for example, with the user requesting only 200 bytes to read, it ended up being 404 for us to go straight into the user's buffer.



This was found in one of our catch cases for our b_read function, mainly when we did not want to read in a total amount of bytes that is larger than the user's total file size.

```
535     //Catch case if what we're reading is more than what's leftover, so we don't go
536     //over the file size
537     printf("COUNT: %d\n", count);
538     if(count + fcbArray[fd].totalBytesRead > fcbArray[fd].totalBytesRead)
539         {
540             printf("b_read() count + fcbArray[fd].totalBytesRead > fcbArray[fd].totalBytesRead\n"
541             count = fcbArray[fd].fileSize - fcbArray[fd].totalBytesRead;
542             printf("count: %d\n", count);
543         }
544
```

Resolution #6:

How we fixed this error was by changing the condition within the if statement, since we mistakenly compared the count + totalBytesRead to just the totalBytesRead, instead of just the overall fileSize. Doing so, it managed to get rid of stack smashing issue for both the cp2l and cat commands.

```
4. wrote parent directory again in b_close() because b_write modifies the fileSize field in the DE so we need to
   save those changes to disk other wise when we call open again when doing cat the newf
bytesRead: 140

close fd: 0
close() fcbArray[fd].fileSize: 540
Prompt >
```

Natalie Yam
Westley Cho
Etinosa Osagie-Amayo
Angelo Lance Quetua
Bryan Yan

Helper Functions

Helper Functions	Author and Description
int getEOF	Bryan
int appendBlocks	Bryan
int findFreeDirEntry	Natalie
void setRootDirParse	Bryan
void setCurrDirParse	Bryan
directoryEntry initDir	Bryan
int isADirectory	Natalie
int parsePath	Everyone
char * cleanPath	Natalie, Westly
char getType	Natalie
FATInitialize	Bryan
int allocateRoot	Bryan
int allocateBlocks	Bryan
int getFATStart	Bryan
int getFreeSpaceStart	Bryan
int freeBlocks	Bryan

void freeFATMap	Bryan
int writeDir	Bryan
directoryEntry * getRoot	Bryan
void setRoot	Bryan
directoryEntry * loadDirectory	Bryan

Helper Function descriptions:

- int getEOF:
    - Used to get the EOF indicator of any given directoryEntry* from our FAT map. Was used as a helper function in allocateBlocks to figure out where to append blocks to.

- int appendBlocks:
    - Used to append blocks to the end of a block chain when a given directoryEntry needed more space.

- void setRootDirParse:
    - Used to set the root directory global in mfs.c so it can be used in all the fs functions.

- void setCurrDirParse:
    - Used to set the current directory global in mfs.c so it can be used in all the fs functions.

- directoryEntry* initDir:
    - Initializes a directory and returns the pointer to the caller.

- FATInitialize:
    - Sets the default values for the FAT map and writes it to disk.

- int allocateRoot:
    - A helper function that allocates the root as opposed to a regular directory. It has the extra functionality of setting the global root directory pointer.

- int allocateBlocks:
    - A helper function that allocates the amount of blocks specified by the caller of the function. One example of its use is in initDir.

- int getFATStart:
    - A getter function that returns the start location of the FAT map on disk.

- int getFreeSpaceStart:
    - A getter function that returns the start location of usable free space.

- int freeBlocks:
    - This function marks the blocks originally used by a directoryEntry as unused so that those blocks can be used for something else later on. This function is called when we are removing a directory or a file.
- void freeFATMap:
    - A helper function that frees all allocated memory for the FAT map for when we want to exit the file system.

- int writeDir:
    - A function that writes a directoryEntry* to disk. This function is used by the fs functions to write directories and files to disk because mfs.c doesn't have access to the FAT map. The writeDir function helps remove the need to read the FAT map from disk, manipulate the map, and write it back to disk in mfs.c.

- directoryEntry * getRoot:
    - A getter function that returns the root directory.

- void setRoot:
    - A setter function for the root directory global.

- directoryEntry * loadDirectory:
    - A helper function that loads a directoryEntry from disk to RAM. It's used in numerous fs functions for when manipulation of a directory is required but one of the most notable uses of the function is its part in parsePath.