CSC415-02

Team: Zombies
Salaar Karimzadeh SFSU ID: 920765141
Jasmine Thind SFSU ID: 920767299
Brandon Cruz-Youll SFSU ID: 902899041
Brian Adams SFSU ID: 921039987

File System Project

https://github.com/CSC415-2022-Spring/csc415-filesystem-BeeSeeWhy

File System Design:

We began the file system project with our file system design. For our Volume Control Block, we planned on a few defines: #define VOLUME_SIZE 100000 (for the size of the volume), #define BLOCK_SIZE 512 (for the size of the block) and #define VOLUME_LOCATION 0 (for the starting location of the volume). Our volume control block would be defined as follows:

```
typedef struct VolumeControlBlock {
        int totalBlocks             // Overall size of the blocks in the volume
        int blockSize;              // The size of our volume block
        int magicNumber             // The type of file
        int rootDirectoryBlock      // location of the root directory
        int initialization;         // Has the VCB been initialized?
        int blockCount;             // Keep track of the block
        int freeBlockCount          // Total number of free blocks
        } VolumeControlBlock;
```

Our Free Space Management would help keep track of the free space. We planned on using a fixed array for free space management. The free space management would keep track of the size of the array and initialization of the free space. It would keep track of the starting point and ending point of the free space. It would also keep track of free location in the freespace. Lastly, it would keep track of block size and bytes.

Our directory entry was planned as follows:

```
typedef struct DirectoryEntry {
        int id;                 // file id
        int entrySize;          // size of file
        int fileBlock;          // file location
        Char isADir;            // is the file a directory?
        DirectoryMD metadata; // file metadata
} DirectoryEntry;
```

```
typedef struct DirectoryMD {
        char* name;             // file name
        int magicNumber;        // file type
        char* dateCreated;      // file creation date
        char* author;           // file author
        char permission[9];     // file permissions
} DirectoryMD;
```

The directory was planned to be a linked list, or an array of various directory entries that contain metadata about the various files owned by a directory. These directory entries would be a struct containing metadata about the entries itself. This metadata included the entry size, entry name, entry id, the entry creation date, the date it was last modified, permissions for various users.

The file system metadata would include: file permission, system recorded times of: creation, last accessed, last modified; and would record the author of the file.

Final Implementation Design:

Our file system design was largely informed by lectures and the reading material in our course work. Our directory system, where directories hold entries that point to other directories or files, tracking the freespace through a structure that can individually address which blocks are free, the extent system in which blocks could be written anywhere and in any order, but then read back sequentially were all discussed or touched on in the class. The hints and suggestions given in class, for example parse path, discussing how contiguous allocation could increase complexity later on also helped inform the decisions we would make during the design process.

One of the more unique design choices for our project was how we handled writing and reading to our file system. One of the largest decisions we had to make was regarding how files should be written to our file system? Should files be one large contiguous block, where we are required to find a series of free blocks? What happens when there is sufficient external fragmentation where we have enough blocks to write but not enough free sequential blocks? What would defragmenting the volume look like? How would files grow?

The more we looked into these answers the more we started to realize that the constraint of requiring sequential blocks for reading and writing created more problems than it solved. Our solution came largely from reading about extents, then looking at other data structures and considering how we can derive a simple and effective solution for concepts we have previously learned. This led us into considering a linked file system.

Our linked file system took many ques from linked lists. One of the best examples being that linked lists don't need to own the next memory address to know where other nodes are because each node simply pointes to the next node. In our file system each block points to the next block. Each block is written with a key which is the last four bytes in every block. This key is an integer that points to the next block that the file is written in. When we write to the file system we simply just need to know where the next free block is, then insert the key in the current block. When we read from our file system we simply pull the key out of the current block which points to the next block.

This design does come with its own set drawbacks though. For one, 4 bytes out of every 512 byte block is being used for identification purposes. This comes out to .7% loss in total volume size assuming no internal fragmentation. Although we remembered from lecture that a standard file system generally has a pretty substantial amount of internal fragmentation. This means that it is likely that many blocks will be written, without enough data to completely fill them. Leaving a difference between the block size (minimum number of bytes that can be allocated to store data) and the size of the data being stored. This would reduce our .7% total loss in volume to a much lower effective loss.

The second obvious drawback is the increased IO associated with having to read a block before being able to read the next block. This means that with our implementation we can only effectively buffer 512 bytes with each read. Had we had more time I would have wanted to explore potentially using negative integers to represent blocks that were able to be contiguous allocated, as in a key of -4 would suggest the next 4 blocks can be read at once as they all belong to this file, and are written in order. In either case the buffers provided in the b_read and b_write were small enough that we believed this wouldn't cause a major bottleneck with the code provided.

Finally, this last drawback is more specific to those who look at the hexdump. When looking at the hex dump, it appears as if the last 4 bytes of every block look "corrupted". As in if you write a text and use the hexdump to look at it, you will be able to read the file, but the last 4 bytes are the key (an integer) which does not translate to a nice character value. You will see however that the expected character is contained within the next block, or the next block the key points to. You will be able to verify that the data was correctly written by either following the pointers, or copying the file back out to the linux file system (cp2l), where the file is reconstructed without the keys. As a quick note to hopefully help determine where the next block to be written will be, our file system will try to write to the next block by default (assuming there is not something already there), so if you write to block 41, and there is nothing in block 42, the next block chosen will be 42.

Overall, I think that this linked file system might not be suitable for very read-heavy environments, but it was an incredibly interesting design to implement and really required us to consider what we learned in our other classes, and how we could apply those concepts in our file system. This design works great in our testing, and can easily grow files around existing files. We included a cat2fs which appends an existing file from the users file system onto the end of an existing file in our file system. This will extend an existing file without the need to alter the existing blocks.

1. MileStone One:

For milestone one, the assignment required us to partition a volume, initialize a volume control block, initialize a freespace structure and the root directory. After the volume has been initialized, we need to initialize and write the volume control block, free space, and root directory. Then run a hexdump to show your SampleVolume has been initialized correctly, and that the volume control block, free space structure and root directory have successfully been written to the SampleVolume and are located in the intended blocks. Finally, we needed to analyze the hex dump and determine what was written.

Volume Control Block:

Our Volume Control Block structure holds the variables that will be written into volume. It includes the total number of blocks, the block size, the start of the free space, the start of the root directory, and the magic number. We also included a prototype of a volume control block initializing function called vcbInit() that contains 5 parameters. Our VCB struct will handle the initialization of variables such as; number of total blocks, size of the blocks, the free block count, all block pointers, and the root directory pointers. Through the usage of pointers it will allow for dereferencing and access to different members in the structure. Our initialization of the VCB in fsInit.c followed the steps of milestone 1 pdf and malloc a block of memory which we set as BLOCK_SIZE as a volume control block pointer. We then called the LBAread() function which takes a buffer count of the block and the starting block number 0. It returns the number of blocks read. Next was to check if our pointer to the structure matched the magic number, to do this we used an if statement and if the magic number does not match it will initialize the values of the volume control block, the free space, and the root directory.

vcbInit(int bSize,int totalB):

This function contains the volume details and initializes the number of blocks, the signature, and the block size. The magic number is defaulted to 16 and is used for identification.The block size represents the size of each logical block in bytes and the total blocks denote the total blocks available in the volume. VCBInit creates the vcb and populates all the required files, and will be written into block one in initFileSystem.

Free Space Management:

Our free space structure needed to be able to keep track of all the free blocks for space allocation to files when they are created. We will need to be able to reuse the space when the files have been deleted. This makes free space management crucial. The free space system maintains a free space list that will keep track of the disk blocks that have not yet been allocated to the directory. We originally planned to use a bitmap array that will keep track of 0's or 1's, and that will tell the state of the block. If a block is not being used then it will be marked with a 'X', if a block is being used then it will be marked with a 'O'. In the free space structure we created an array of characters that are defaulted to 'O'. We ended up implementing a char array, with each character index representing whether a was free.

This is objectively an inferior implementation from the bitmap. Using an array of characters over a bitmap requires 7 more bits per block represented. This means that in our 10,000,000 byte volume, comprised of 19,531 blocks (512 bytes per block), our implementation takes 39 blocks (38.1 but rounded up because due to the nature of internal fragmentation) to record the current usage of every block, whereas a bitmap would require only 5 blocks to record this same information. Our implementation can record the value of 512 logical blocks per block consumed, where the bitmap can record 4,096 logical blocks per block consumed. At any scale the bitmap will outperform the array, and as the total block count increases the bitmap becomes even more efficient compared to our array. If we had more time, and tests written to ensure all the consumers of our freespace abstractions functioned after a large change we would have changed our implementation. Given the time constraints, and technical constraints (maximum of 10,000,000 byte volumes) we felt that the array, while not optimal, was sufficient enough to be considered a low priority for refactoring.

This array is then written to the Sample Volume, once the VCB has been written. In the freespace structure we keep track of the block size, volume size, the size of the array, the location of that array(which will be initialized at 1) and the free space location. We initialized the bits using malloc by multiplying the array size by the block size.

void initFreeSpace():

This function will initialize the free space in the volume control block. We will mark the spaces off the blocks in the file system when started. If a block is free we will mark it with an O, if a block is being used then we will mark it with an X. The freespace is tracked with an array, each indice mapping to a single block. initFreeSpace is incharge of writing the fsArray into the LBA.

Directory:

Directories are the foundation for a user-friendly filesystem. We could certainly have a series of Directory Entries in one very long array that simply pointed to files, and this would be more efficient for the computer as we are not incurring the overhead of loading a directory from the file system, locating the directory entry for the next directory we want to traverse to, pulling that directory out of the file system, and repeating until we reach the directory or file we were searching for, but this would make for an incredibly poor user experience. Being able to logically group entries, and "nest" logically grouped files "inside" other logical grouping of files through a directory system strikes a good middle ground of being able to maintain a flat record of files that a computer is efficient at parsing, while offering a reasonable way for people to interact with that file system that simulates containers for files.

Our directories facilitate this exact function, there is one root directory, that contains directory entries which contain data (including where to find files and/or other directories) about all files/directories "logically grouped within" this directory. We can then create other directories and a directory entry that points at this new directory, then insert this directory entry into the directory entry array within the parent directory. This way the newly created directory can be traversed to through the matching the directory name in the parents directory entry array.

The directory entry structure holds everything that is needed to identify the file that will be described in the entry–including, but not limited to, the file id, name, size of the file and the volume block the file is located. For the initialization of the root directory, memory is allocated for a directory structure object with the name "root". The "." and ".." are added to the root directory. All other entries are given a file id of -1 and a name of "" to show they are open entries.

fsDir* initRootDir():

This function will initialize the root directory with default values, (references itself for its current and parent directory entries, sets the creating date, author etc) and is returned as a struct that can be written into block number one. The block location is hard coded to ensure that the root directory is always able to be found. This root

directory is the beginning for every single file/directory lookup, having a fixed, and deterministic access point is a requirement for a functioning file system.

fsDir* makeDir(const char* name, int blockLocation, fsDirEntry parentDirEntry):

This function will create a directory, provide default values (creates parent directory entry, current directory entry, sets the name, creation date, file location etc). This function differs from initRoot in that the parent directory does not refer to itself, its invocation happens when the user intends to create a file, and not by the file system on when the file system is created..

Milestone One Issues:

1. We had a hard time conceptualizing how to run the code and check if everything is writing to the hexdump. We knew how to get the output for the hexdump, as it was shown in class, but at first we did not know how to initialize the hexdump with the vcb, directory, and freespace.
    a. We finally figured out that fslow.o had all the functions that we needed to be able to write our file system to the hexdump, which allowed us to finally see our initialization in the dump.

2. It was difficult for us to wrap our brains around this project as it was the first time for all of us creating a file system, but I can assume it was the same for every group.

    a. Rewatching the lectures and lecture recordings helped us with understanding what needed to be done. Reading the Milestone requirements multiple times and making a

3. We had issues with scheduling and working issues as everyone was busy with other classes. A lot of the time we intended to work together but people needed to study for midterms or had other projects

    a. We tried our best to work around these projects and study time, trying our best to at least get a couple of hours in to work together as a lot of the functions needed different parts from different files.

4. We expended an extreme amount of time to actually create and implement a fsLow.c for LBAWrite and LBARead, but soon realized that an fsLow.o with every functionality was given to us.

a. On our part we were really confused at first, but went through the github again and read through each instruction. We then realized that a fsLow.o has each function that needed to be called. But we took an extra amount of time to try and create a partition.

5. It took us a while to understand the changes to be made to the MakeFile. Once we figured out the files were not linking together during the compiling, we remembered the additional objects to add to the MakeFile.

2. Milestone Two

Int fs_mkdir(const char *pathname, mode_t mode):

This function will create a new directory with the name of a given path, that will update the parent directory with information into the disk. This function is responsible for ensuring that the path is valid, and a directory can be made at the specified path. This involves traversing the path, loading the parent directory, verifying that the parent directory is indeed a directory, checking to see if the directory array is full, then creating a directory and directory entry, writing that newly created directory into the file system, and appending the directory entry into the parents directory entry array then saving the updated parent directory back into the block it was loaded from.

Int fs_rmdir(const char *pathname):

This function will remove a directory with a valid path given. If an invalid path is given an error is thrown. Removing the directory is a lot simpler than one would imagine. The process to remove a directory includes finding the correct directory to remove, again this includes verifying the path is correct, that the target is a directory, that the directory can be removed (this includes checking to see if the directory itself is empty, which is a requirement to remove the directory). Once all those guards have been passed we simply remove the directory entry from the parent, update the free space to mark the blocks the directory was previously written to are now free, then save the parent directory back into the block it was occupying to update it. There is no need to manually overwrite the blocks that the directory was occupying. Removing the entry and marking the space as writable is sufficient.

fdDir *fs_opendir(const char *name):

This function will open a directory by the path provided, and will return a pointer that stores the info of the directory. Just like mkdir we ensured that the path is valid, and target is a directory that can be opened.

Struct fs_diriteminfo *fs readdir(fdDir *dirp):

This function will return a pointer to a structure that stores the information of each directory entry or will return a NULL if the end of the directory is met. This traverses the directory entry array and appends data about each entry into the directory info buffer passed into the function, then returns a pointer to that struct. We use the directory info buffer to store the current index of the directory entry array we need to search for. This allows us to maintain state between readdir calls without creating a global object or altering the function signature by passing in an external structure to track the state.

Int fs_closedir(fdDir *dirp):

Deallocates memory and free memory for the open directory. Frees the opened directory pointer. We deallocate the directoryInfo struct itself as well as the dirp pointer.

char *fs_getcwd(char *buf, size t size):

This function will copy an absolute pathname of the current working directory to an array using a pointer called buf. We store the current path in a global variable called currentPath. This is updated in setcwd, meaning when getcwd is called we already know that we are in a valid directory, and that the current path is the current location. This means all we need to do is copy the currentPath to the buffer provided, then return that buffer.

Int fs_setcwd(char *buf):

This function will take a path and check to see if that path is valid. We then use the parsePath function to tokenize the path, splitting on /. We then check to see if the token is an existing directory entry in the current directory. If it is we append it to the current global path string, if not we return an error. We repeat this process until we have exhausted tokens or an invalid path is detected.

Int fs_isFile(char *path):

This function will take a given path, if the path is valid we will load the parent directory, then check to see if it is a file in the directory. Files are denoted by not being a directory (a ‘F’ for the isDir field in the entry). If the entry exists and is not a directory we then return 1;

Int fs_isDir(char *path):

This function will take a given path, checks to see if the path is valid. We will check to see if the path is an existing directory or not. These directories are marked as T. This behaves identically to isFile, with the exception that we are checking to see if the isDir field equals ‘T’.

int fs_delete(char *filename):

This function will take a given path and will check if that path is valid using the parsepath functions. If the path is valid we will load the parent directory of the file, and check that the file is in the current list of directory entries, and is removable. If it is, we then remove the directory entry, shifting down the following directory entries to take its place, and mark the blocks the file originally occupied as being free. We then save the parent directory back into the block it was pulled from to update it. Since we used a linked files system where each block contains a key to the next written block we need to traverse the file, marking the free blocks individually as we traverse the file until we reach the end.

int fs_stat(const char *path, struct fs stat *buf):

This function will take a given path and find the name of the directory entry. Will also fill out the data files of the fs_stat structure. This data includes date created, last date modified, file size etc. Just like many of the other functions we check to see if the path is valid, then pull the entry from the parent directory.

parentPath* getParentPath(const char* path):

This function will take a full path of a file that a user would like to access, an example being dir1/dir2/file. This function then finds the final / and separates off the last file/directory, and the remaining path. This data is then returned in a struct where the file name of the path, and the path to the parent are available to use. This is an important utility because we are often needing to traverse to the parent directory in order to update or remove the requested file/directory.

fs_Path* parsePath(char* path):

This function will take the current path to the resource, as well as the directory entry of the resource. Parse path traverses the given path and either finds directory entry the resource requested, or returns a null pointer. This is a very important

abstraction as the majority of interactions requires manipulating the resources directory entry. This allows us to reach that entry in a uniform manner throughout our application.

parentPath* relPath(const char* currentDir, const char* relPath):

This function was a last minute addition, ideally this should have been included in parsePath. When initially writing this application we hadn't considered that relative paths can include more than just the relative file or directory name (cd dir vs cd dir1/dir2). This is a shim used to adapt the second example into our existing code that only identifies relative paths as the first example. This simply appends the current directory onto the relative path given then returns an object with the full path. We can then consume the relative path like an absolute path.

Milestone Two Issues:

1. For most of this milestone the scope of the entire project was getting very tricky. We had issues wrapping our brain around what functions we needed to pull from and started to over complicate everything.
    a. Brian helped a lot with the functionality of the entire project, he understood what we needed and how to get it done when we were stuck. His parsepath function helped a lot throughout the mfs file.
2. Functionality for the free space had to be changed as we needed to keep track of where these blocks were getting filled and check if that block was free to write to.
    a. Created a couple functions throughout milestones two and three that allowed us to check for a free block, mark a free block and get a free block.
3. Creating a uniform way to manage the path. One of our larger pain points came from a pretty large oversight in this milestone. We treated absolute paths and relative paths differently in each function that needs to consume a path. This means instead of passing the path into a function that always returns the absolute path, we check if the path passed to the function is absolute, if so we give it to parsePath, if not we know the current directory we are in is valid so we just search to see if the entry exists. This creates a lot of repetitive logic and messy code. We certainly save on our io operations, but this comes at the cost of very brittle code being inserted at the top of each operation that needs to consume a path.
    a. We wrote a couple functions to abstract away this logic, and ended up creating a function that returns the absolute path from the relative path. We still have to determine whether or not the path is absolute or relative in each consuming function but found this to be an adequate compromise

between a full refactor, and leaving it as is. Had the scope of this application been larger, or had I known that this code was going to be maintained I would have completely refactored it, and provided a clean abstraction to use. With that said, the lack of tests written and time constraints we felt led us to be apprehensive about a full refactor, and instead choose the more pragmatic approach of writing a smaller adapter.

4. Milestone 3

B_io_fd b_open(char *filename, int flags):

This function is an interface to open a buffered file in the file system. We will check for a valid parent path, that can handle any relative or absolute path and parse through the file. We will also check to see if that file exists in the directory if not we set up the fcb.

Int b_seek(b_io_fd fd, off_t offset, int whence):

This function was the largest point of confusion for us. It was unclear exactly when this would get called, or how it would get called. We understood its intention and what it does (move the cursor of the file being read or written to, to a specific byte), but looking over fsshell it appeared that none of the existing commands would call this function, nor would any existing commands change the flags passed to b_open beyond what was specified in cp2fs and cp2l. In either case our implementation of b_seek is being used by our append function (we added a cat2fs command that takes a file on the linux file system, and appends the content to an existing file on our filesystem, then saves that concatenated file). The implementation was straightforward as it simply seeks to the end of a file, loading the last block into the readBuffer, and setting the bytes read to reflect the end of the file. This allows us to execute the write function as if we are already in the middle of writing a file from a buffer.

Int b_write(b_io_fd fd, char* buffer, int count):

b_write is incharge of reading from a buffer, filling a buffer that has been malloced on the file descriptor, then saving the block into the LBA. To do this b_write must first determine which block is free. To do this b_write calls the getFreeSpace function which searches the free store for free blocks. Due to our design b_write will also need to know the next block we write to as this is going to be the key saved in the last four bytes of the block. To facilitate this we have a prevKey field on the file

descriptor that stores the location of the key from the previous block to provide the location for the current block to write to.

This function has three states that it can fall into. The first state will be able to fit the entire buffer into the read block, having no overflow. The second state will fill the read buffer, keep a key with the location of the write(which will allow us to keep track of the filled blocks) and then reset the buffer to refill the remaining overflow. The third state will check if it can or cannot fill the entire buffer. If we do have room for the entire buffer we repeat the steps of the second state.

Int b_read(b_io_fd fd, char* buffer, int count):

b_read is largely the inverse of b_write, where we are loading the blocks from the LBA, and reading them into the buffer. We get the key of the next block to read from by extracting the key from the current block. We repeat this process of reading blocks, filling the buffer, committing filled buffers, then reading blocks until we have reached the end of the file. The end of the file is determined by the totalByteRead count, equalling the file size. This ensures that we don't overread the last block, or try to pull keys that do not exist.

Void b_close(b_io_fd fd):

Be close releases the file descriptor back to being in a free state, by freeing the allocated buffer and setting it back to null. There is also various clean up for parts of the file descriptor that was used in the various read write operations. This includes freeing the read buffer, creating directories entries for files that were written and freeing the parent directory that was opened in open.

Milestone Three Issues:

1. The primary issue we faced with this milestone involved rushing through the read and write functions. Instead of defining the various states that the read and write operations can be in, then working through them in an organized manner, our initial iteration ended up being a chain of various conditionals, with additions to the byte counters peppered throughout. This made it incredibly difficult to debug and make sense of what was being incremented where. We wound up with some discrepancies in the block count, and total byte count and occasional buffer overflows due to this disorganized approach.
    a. The solution was to do a complete rewrite of these functions. Instead of jumping into reading and writing to buffers we considered the various

states that the read and write functions can be in. Then individually wrote logic for each, making sure to pay attention to when and where values were being incremented. This more methodical approach ended up being functional, easier to reason about code. This resolved all of our count and buffer issues. While there might be a time and place for quickly written code, we can say with confidence that core operations like reading and writing are not the correct place.

2. The second issue was b_seek. There was a lot of confusion on where this fit into the larger project. It wasn't clear where this function would be called. After looking over all the existing commands it was clear that none of the commands in fsshell used it. There was the additional confusion about the breath of flags being passed to b_open. The flags passed from the cp2l and cp2fs made sense but just like b_seek we were not sure when and how additional flags would be passed.
    a. In the end we wrote b_seek to aid in seeking specific parts of our file. It is used in our cat2fs command, where b_seek moves prepares the specified file descriptor state to append data to the end of the file. We also made use of various flags to give our open and close functions more context about what operations are going to be required.

Final Core Functions of the FileSystem

fsDir* makeDir(const char* name, int blocklocation, fsDirEntry parentDirEntry):

This function will create a directory in the volume control block, each directory entry is marked off as T. This function will pass the name of the directory, the location that it has been written to, and the directory entry of the parent.

Void addDirEntryFromDir(fsDir targetDir, fsDir* sourceDir, int targetIndex):

This function will allow us to add a directory to any of our directory entries in the file system. The parameters passed check for adding a directory to our directory entry, the location of the source directory entry data, and where you would like to write the source directory data to. This function will help us make a directory in mkdir, adding an entry to the directory.

fsDirEntry* findDirEntry(fsDir* src, char* dirname):

This function will look for a directory entry in our directory allowing us to search for directory entries needed in the file system. If an entry is not located in the directory then we will return NULL. This function will be used in our b_io open allowing us to check if a file exists or not.

Int fileNameExistsInDirEntry(fsDir* src, char* dirname):

This function is used in the b_io open to check if that file name exists in the directory entry. We pass the src to search for the directory

Int rmDirEntry(fsDir* src, char* dirname):

This function is used to remove a directory entry in the directory. We pass the src to search for the directory entry in the directory and the directory name.

fsDir* loadDirFromBlock(int blocklocation):

This function will check if the block you are reading from holds a directory. We use this in our b_io open to check if the file exists in the directory already, and use it in b_io close for deallocation.

Void setRead(fsDirEntry* targetEntry, int status):

This function allows us to change the directory permissions, allowing for if we want to set a directory to read.

Void setWrite(fsDirEntry* targetEntry, int status):

This function allows us to change the directory permissions, allowing for if we want to set a directory to write.

Void setDelete(fsDirEntry* targetEntry, int status):

This function allows us to change the directory permissions, allowing for if we want to set a directory to delete.

void markUsedSace(freeData file):

This function will mark a block as used in the vcb. When we look at the volume each block that is taken will be marked with an ‘X’.

Void markUsedSpaceByBlock(int start, int numberOfBlocks):

This function will do the same as MarkUsedSpace but will mark the volume block by block. We use the same symbol if the block is taken, this will be marked with an ‘X’

Void markFreeSpace(int location, int size):

This function will mark a location in the vcb as free. When looking at the volume each block will be marked with an ‘0’ stating that block is free.

freeData getFreeSpace(int blockAmount):

This function will check to see if there is free space in the vcb. Will pass the blocks needed by the user and return the amount asked for. This is used in functions to be able to mark blocks off in the freespace, allowing us to make directories, or write to blocks in the volume.

char* getDataFromBlock(char* buffer, int buffersize):

This function will allow us to check what has been written to the block in the LBA. This helper function is used with getKeyFromBlock to fetch all the data in a block that is not the key. The last 4 bytes of our blocks are used for keys which identify the next block to read. This helps us grab all the data in the buffer up until those 4 bytes.

Int getKeyFromBlock(char* buffer, int bufferSize):

This function accepts a buffer and the size of the buffer. The function then reads the last four bytes of the buffer and casts it to an int. This integer represents the next block to read. There are no safeguards with this function so it is up to the caller to verify that this function is passed a block that came from the LBA and has been written correctly with a key already inserted.

Int writeKeyToBuffer(char* buffer, int bufferSize, int key):

writeKeyToBuffer is the inverse of getKeyFromBlock. This function takes in a buffer, its size and an integer to write to the end of the buffer. This key will be used to identify the next block to read. Again just like getKeyFromBlock this function has no

safeguards, and will overwrite the last 4 bytes of the buffer if the caller wrote data they had intended to keep to the full buffer.

How our Driver Program Works:

The implementation for our driver program is located in fsshell.c. This has the main function to run the initialization for the file system as well as all the commands to take user input, allowing for output of any information needed by the user. In the main function of fsshell.c the first thing that is called is the startPartitionSystem() function which will pass the file name, volume size, and the block size from the command line arguments. The next function that is invoked is the initFileSystem() which will load our volume control block, as well as initialize the freespace, write a location for the vcb as well as write a location for our directory. After the initialization of our file system, the driver program will be able to read from the volume and access data needed at a memory level. Next, the driver program will be able to prompt the user with an interactive interface that will allow for commands to be entered, each command will have its own set of functionality that will output information of the file system or process files in the directory. Finally, the driver program can exit the shell by calling the function exitFileSystem() and closePartitionSystem() in the main function. Overall, we have a simple file system that can perform numerous tasks, and be able to process those files.

In our driver program, fsshell.c allows the user to use various implementations of the interfaces provided such as b_write, b_read, mkdir and rmdir. fsshell also provides a lot of set up for these functions. This includes allocating buffers, providing pointers to write data to, and verifying if entries are directories through isDir and isFile for some commands. Fsshell also provides some of the implementation for specific commands like cp that will generate two separate file descriptors, one for the source and the other for the file system. Cp will then pass those file descriptors into b_read and b_write as well as a buffer which facilitates the cp command from our already created read and write. Another function we can look at such as, cmd_md, which will pass the file or relative path requested to make a directory, as well as the file permissions that should be associated with that created directory, in this case 0777 which is a read, and write permission in linux. After these are passed in we will create a new directory that has the name of the last token given in the path, which will then update the parent directory into disk.

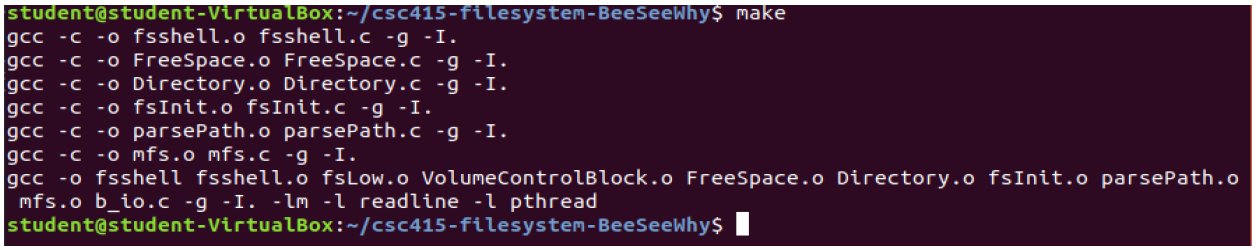
A table of who worked on which components:

Volume Control Block	Jasmine Thind
Free Space	Salaar Karimzadeh/Brian Adams
Directory System	Brian Adams/ Brandon Cruz-Youll
Write Up Milestone 1	Zombies Team
fsInit	Zombies Team
Debugging	Brian Adams
Parse Path Utility	Brian Adams
fs_isFile/fs_isDir	Brian Adams
fs_delete	Brian Adams/ Brandon Cruz-Youll
fs_rmdir	Brian Adams/ Brandon Cruz-Youll
fs_mkdir	Brian Adams/ Salaar Karimzadeh
fs_opendir	Brian Adams
fs_readdir	Brian Adams/ Salaar Karimzadeh
fs_closedir	Brian Adams
fs_getcwd	Brian Adams
fs_setcwd	Brian Adams
fs_mv	Brian Adams
fs_stat	Brian Adams
b_open	Brian Adams/ Jasmine Thind

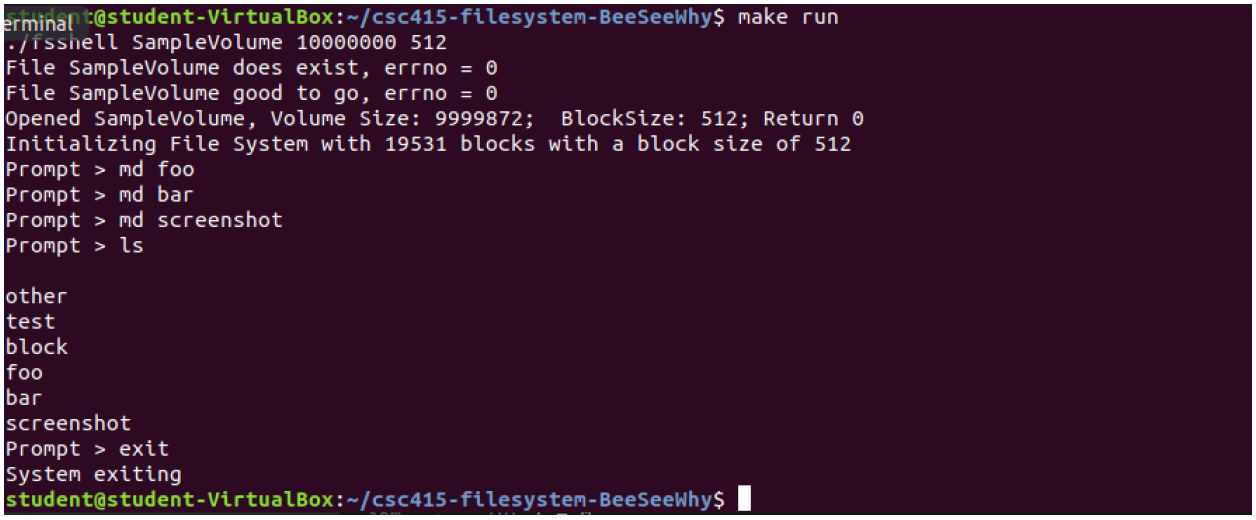
b_close	Brian Adams/ Salaar Karimzadeh
b_read	Brian Adams/Salaar Karimzadeh
b_write	Brian Adams
b_seek	Brian Adams
Code comments	Salaar Karimzadeh/Brian Adams
Final Write Up	Brandon Cruz-Youll/ Salaar Karimzadeh/ Brian Adams

Screen Shots:

Make



ls & md



Hexdump for ls & md

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ Hexdump/hexdump.linux --count 2 --start 3 SampleVolume
Dumping file SampleVolume, starting at block 3 for 2 blocks:

000600: D0 A0 65 62 00 00 00 00  D0 A0 65 62 00 00 00 00 | Peb....Peb....
000610: D0 A0 65 62 00 00 00 00  26 10 00 00 75 73 65 72 | Peb....&...user
000620: 00 58 58 58 58 58 58 58  58 58 58 58 58 58 58 4F | .XXXXXXXXXXXXXO
000630: 72 77 64 00 4F 4F 4F 4F  FF FF FF FF 66 6F 6F 00 | rwd.OOOO◆◆◆◆foo.
000640: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000650: 00 00 00 00 00 00 00 00  00 00 00 00 02 00 00 00 | ................
000660: 5F 00 00 00 54 00 00 00  00 00 00 00 00 00 00 00 | _...T...........
000670: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000680: 00 00 00 00 75 73 65 72  00 00 00 00 00 00 00 00 | ....user........
000690: 00 00 00 00 00 00 00 00  72 77 64 00 00 00 00 00 | ........rwd.....
0006A0: FF FF FF FF 62 61 72 00  00 00 00 00 00 00 00 00 | ◆◆◆◆bar.........
0006B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006C0: 00 00 00 00 02 00 00 00  61 00 00 00 54 00 00 00 | ........a...T...
0006D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006E0: 00 00 00 00 00 00 00 00  00 00 00 00 75 73 65 72 | ............user
0006F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000700: 72 77 64 00 00 00 00 00  FF FF FF FF 73 63 72 65 | rwd.....◆◆◆◆scre
000710: 65 6E 73 68 6F 74 00 00  00 00 00 00 00 00 00 00 | enshot..........
000720: 00 00 00 00 00 00 00 00  00 00 00 00 02 00 00 00 | ................
000730: 63 00 00 00 54 00 00 00  00 00 00 00 00 00 00 00 | c...T...........
000740: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000750: 00 00 00 00 75 73 65 72  00 00 00 00 00 00 00 00 | ....user........
000760: 00 00 00 00 00 00 00 00  72 77 64 00 00 00 00 00 | ........rwd.....
000770: FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ◆◆◆◆............
000780: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000790: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007A0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007C0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007D0: 00 00 00 00 00 00 00 00  FF FF FF FF 00 00 00 00 | ........◆◆◆◆....
0007E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```

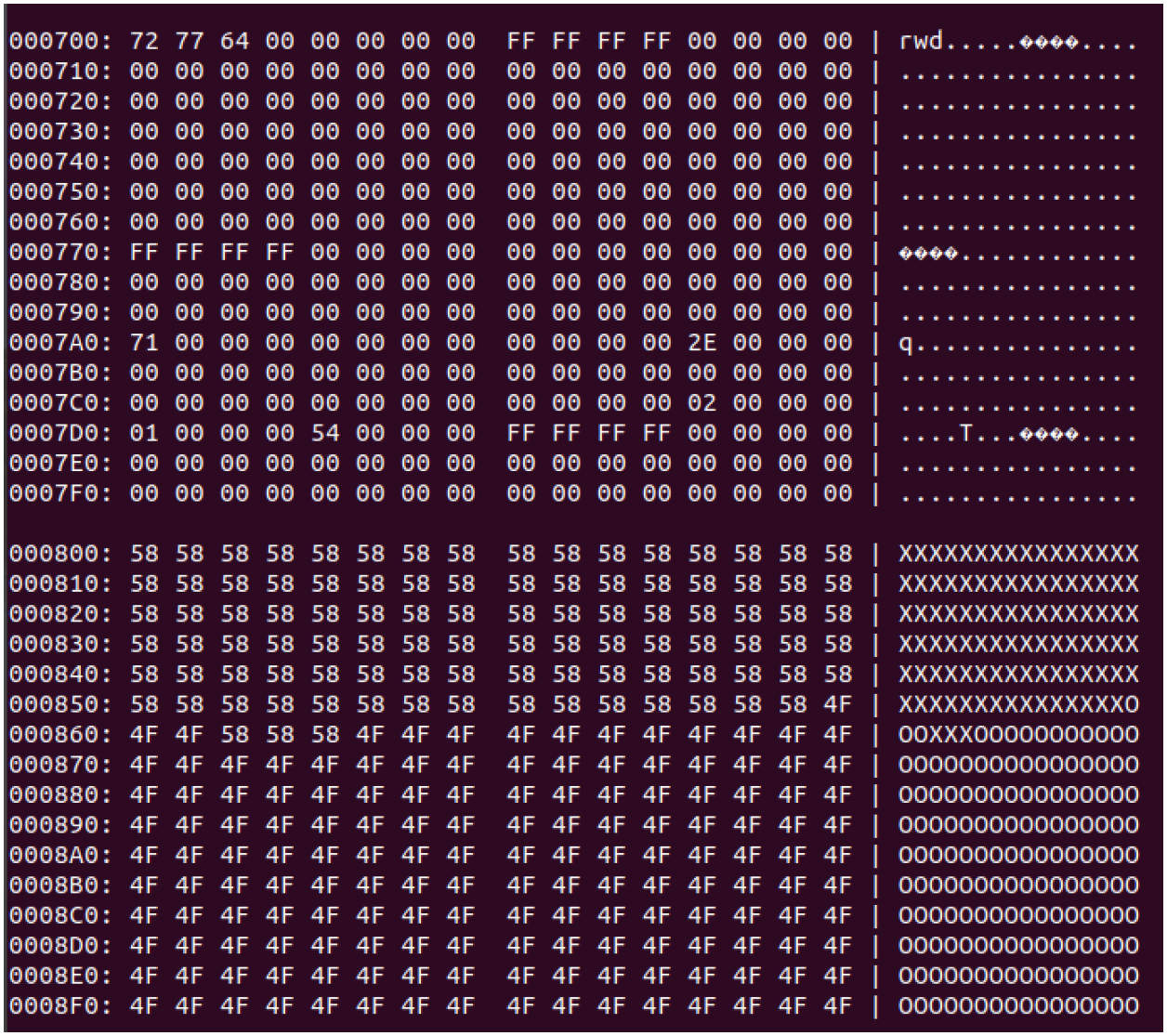
```
000800: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000810: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000820: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000830: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000840: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000850: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000860: 58 58 58 58 58 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | XXXXXOOOOOOOOOOO
000870: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000880: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000890: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008A0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008B0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008C0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008D0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008E0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008F0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO

000900: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000910: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000920: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000930: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000940: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000950: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000960: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000970: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000980: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000990: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009A0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009B0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009C0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009D0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009E0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009F0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
```

rm

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > ls

other
test
block
foo
bar
screenshot
Prompt > rm foo
Prompt > ls

other
test
block
bar
screenshot
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

Hexdump for rm

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ Hexdump/hexdump.linux --count 2 --start 3 SampleVolume
Dumping file SampleVolume, starting at block 3 for 2 blocks:

000600: D0 A0 65 62 00 00 00 00  D0 A0 65 62 00 00 00 00 | Peb....Peb....
000610: D0 A0 65 62 00 00 00 00  26 10 00 00 75 73 65 72 | Peb....&...user
000620: 00 58 58 58 58 58 58 58  58 58 58 58 58 58 58 4F | .XXXXXXXXXXXXXO
000630: 72 77 64 00 4F 4F 4F 4F  FF FF FF FF 62 61 72 00 | rwd.OOOO....bar.
000640: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000650: 00 00 00 00 00 00 00 00  00 00 00 00 02 00 00 00 | ................
000660: 61 00 00 00 54 00 00 00  00 00 00 00 00 00 00 00 | a...T...........
000670: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000680: 00 00 00 00 75 73 65 72  00 00 00 00 00 00 00 00 | ....user........
000690: 00 00 00 00 00 00 00 00  72 77 64 00 00 00 00 00 | ........rwd.....
0006A0: FF FF FF FF 73 63 72 65  65 6E 73 68 6F 74 00 00 | ....screenshot..
0006B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006C0: 00 00 00 00 02 00 00 00  63 00 00 00 54 00 00 00 | ........c...T...
0006D0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0006E0: 00 00 00 00 00 00 00 00  00 00 00 00 75 73 65 72 | ............user
0006F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
```



```
000700: 72 77 64 00 00 00 00 00  FF FF FF FF 00 00 00 00 | rwd.....▯▯▯▯....
000710: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000720: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000730: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000740: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000750: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000760: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000770: FF FF FF FF 00 00 00 00  00 00 00 00 00 00 00 00 | ▯▯▯▯............
000780: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
000790: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007A0: 71 00 00 00 00 00 00 00  00 00 00 00 2E 00 00 00 | q...............
0007B0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007C0: 00 00 00 00 00 00 00 00  00 00 00 00 02 00 00 00 | ................
0007D0: 01 00 00 00 54 00 00 00  FF FF FF FF 00 00 00 00 | ....T...▯▯▯▯....
0007E0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................
0007F0: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 | ................

000800: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000810: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000820: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000830: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000840: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 58 | XXXXXXXXXXXXXXXX
000850: 58 58 58 58 58 58 58 58  58 58 58 58 58 58 58 4F | XXXXXXXXXXXXXXXO
000860: 4F 4F 58 58 58 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOXXXOOOOOOOOOOO
000870: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000880: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000890: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008A0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008B0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008C0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008D0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008E0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0008F0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
```

```
000900: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000910: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000920: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000930: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000940: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000950: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000960: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000970: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000980: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
000990: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009A0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009B0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009C0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009D0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009E0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
0009F0: 4F 4F 4F 4F 4F 4F 4F 4F  4F 4F 4F 4F 4F 4F 4F 4F | OOOOOOOOOOOOOOOO
```

Cd

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > md bar/bar
Prompt > md bar/ber
Prompt > md bar/bir
Prompt > ls

other
test
block
bar
screenshot
Prompt > cd bar
Prompt > ls

bar
ber
bir
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

pwd

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > pwd
/
Prompt > cd bar
Prompt > pwd
/bar
Prompt > cd bar
Prompt > pwd
/bar/bar
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

history

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > ls

other
test
block
bar
screenshot
Prompt > cd bar
Prompt > cd ber
Prompt > cd ..
Prompt > md ber
Error: Duplicate Dir Name
Prompt > ls

bar
ber
bir
Prompt > history
ls
cd bar
cd ber
cd ..
md ber
ls
history
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

help

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > help
ls      Lists the file in a directory
cp      Copies a file - source [dest]
mv      Moves a file - source dest
md      Make a new directory
rm      Removes a file or directory
cp2l    Copies a file from the test file system to the linux file system
cp2fs   Copies a file from the Linux file system to the test file system
cd      Changes directory
pwd     Prints the working directory
history Prints out the history
help    Prints out help
cat2fs  Appends a file from the Linux file system to a file in the test file system
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

cp2fs

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > cp2fs test.txt test_copy.txt
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

hexdump for cp2fs

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ Hexdump/hexdump.linux --count 2 --start 106 SampleVolume
Dumping file SampleVolume, starting at block 106 for 2 blocks:

00D400: 69 63 6F 6E 20 6F 66 20  53 61 69 6E 74 20 50 65 | icon of Saint Pe
00D410: 74 65 72 20 61 6E 64 20  53 61 69 6E 74 20 50 61 | ter and Saint Pa
00D420: 75 6C 2C 20 61 73 6B 69  6E 67 20 70 65 72 6D 69 | ul, asking permi
00D430: 73 73 69 6F 6E 2C 0A 61  73 20 61 20 6D 61 72 6B | ssion,.as a mark
00D440: 20 6F 66 20 74 68 65 69  72 20 67 72 61 74 69 74 |  of their gratit
00D450: 75 64 65 20 66 6F 72 20  74 68 65 20 62 65 6E 65 | ude for the bene
00D460: 66 69 74 73 20 68 65 20  68 61 64 20 63 6F 6E 66 | fits he had conf
00D470: 65 72 72 65 64 20 6F 6E  20 74 68 65 6D 2C 0A 74 | erred on them,.t
00D480: 6F 20 62 75 69 6C 64 20  61 20 6E 65 77 20 63 68 | o build a new ch
00D490: 61 6E 74 72 79 20 74 6F  20 74 68 65 20 63 68 75 | antry to the chu
00D4A0: 72 63 68 20 61 74 20 74  68 65 69 72 20 6F 77 6E | rch at their own
00D4B0: 20 65 78 70 65 6E 73 65  20 69 6E 20 68 6F 6E 6F |  expense in hono
00D4C0: 72 0A 6F 66 20 50 65 74  65 72 20 61 6E 64 20 50 | r.of Peter and P
00D4D0: 61 75 6C 2C 20 68 69 73  20 70 61 74 72 6F 6E 20 | aul, his patron
00D4E0: 73 61 69 6E 74 73 2E 20  49 6E 20 61 6E 6F 74 68 | saints. In anoth
00D4F0: 65 72 20 70 6C 61 63 65  20 74 68 65 20 77 6F 6D | er place the wom
```

```
00D500: 65 6E 20 77 69 74 68 0A  69 6E 66 61 6E 74 73 20 | en with.infants
00D510: 69 6E 20 61 72 6D 73 20  6D 65 74 20 68 69 6D 20 | in arms met him
00D520: 74 6F 20 74 68 61 6E 6B  20 68 69 6D 20 66 6F 72 | to thank him for
00D530: 20 72 65 6C 65 61 73 69  6E 67 20 74 68 65 6D 20 |  releasing them
00D540: 66 72 6F 6D 20 68 61 72  64 0A 77 6F 72 6B 2E 20 | from hard.work.
00D550: 4F 6E 20 61 20 74 68 69  72 64 20 65 73 74 61 74 | On a third estat
00D560: 65 20 74 68 65 20 70 72  69 65 73 74 2C 20 62 65 | e the priest, be
00D570: 61 72 69 6E 67 20 61 20  63 72 6F 73 73 2C 20 63 | aring a cross, c
00D580: 61 6D 65 20 74 6F 20 6D  65 65 74 0A 68 69 6D 20 | ame to meet.him
00D590: 73 75 72 72 6F 75 6E 64  65 64 20 62 79 20 63 68 | surrounded by ch
00D5A0: 69 6C 64 72 65 6E 20 77  68 6F 6D 2C 20 62 79 20 | ildren whom, by
00D5B0: 74 68 65 20 63 6F 75 6E  74 E2 80 99 73 20 67 65 | the count’s ge
00D5C0: 6E 65 72 6F 73 69 74 79  2C 20 68 65 20 77 61 73 | nerosity, he was
00D5D0: 0A 69 6E 73 74 72 75 63  74 69 6E 67 20 69 6E 20 | .instructing in
00D5E0: 72 65 61 64 69 6E 67 2C  20 77 72 69 74 69 6E 67 | reading, writing
00D5F0: 2C 20 61 6E 64 20 72 65  6C 69 67 00 6A 00 00 00 | , and relig.j...
```

```
00D600: 69 6F 6E 2E 20 4F 6E 20  61 6C 6C 20 68 69 73 20 | ion. On all his
00D610: 65 73 74 61 74 65 73 20  50 69 65 72 72 65 0A 73 | estates Pierre.s
00D620: 61 77 20 77 69 74 68 20  68 69 73 20 6F 77 6E 20 | aw with his own
00D630: 65 79 65 73 20 62 72 69  63 6B 20 62 75 69 6C 64 | eyes brick build
00D640: 69 6E 67 73 20 65 72 65  63 74 65 64 20 6F 72 20 | ings erected or
00D650: 69 6E 20 63 6F 75 72 73  65 20 6F 66 20 65 72 65 | in course of ere
00D660: 63 74 69 6F 6E 2C 0A 61  6C 6C 20 6F 6E 20 6F 6E | ction,.all on on
00D670: 65 20 70 6C 61 6E 2C 20  66 6F 72 20 68 6F 73 70 | e plan, for hosp
00D680: 69 74 61 6C 73 2C 20 73  63 68 6F 6F 6C 73 2C 20 | itals, schools,
00D690: 61 6E 64 20 61 6C 6D 73  68 6F 75 73 65 73 2C 20 | and almshouses,
00D6A0: 77 68 69 63 68 20 77 65  72 65 20 73 6F 6F 6E 0A | which were soon.
00D6B0: 74 6F 20 62 65 20 6F 70  65 6E 65 64 2E 20 45 76 | to be opened. Ev
00D6C0: 65 72 79 77 68 65 72 65  20 68 65 20 73 61 77 20 | erywhere he saw
00D6D0: 74 68 65 20 73 74 65 77  61 72 64 73 E2 80 99 20 | the stewards’
00D6E0: 61 63 63 6F 75 6E 74 73  2C 20 61 63 63 6F 72 64 | accounts, accord
00D6F0: 69 6E 67 20 74 6F 0A 77  68 69 63 68 20 74 68 65 | ing to.which the
```

```
00D700: 20 73 65 72 66 73 E2 80  99 20 6D 61 6E 6F 72 69 |  serfs’ manori
00D710: 61 6C 20 6C 61 62 6F 72  20 68 61 64 20 62 65 65 | al labor had bee
00D720: 6E 20 64 69 6D 69 6E 69  73 68 65 64 2C 20 61 6E | n diminished, an
00D730: 64 20 68 65 61 72 64 20  74 68 65 0A 74 6F 75 63 | d heard the.touc
00D740: 68 69 6E 67 20 74 68 61  6E 6B 73 20 6F 66 20 64 | hing thanks of d
00D750: 65 70 75 74 61 74 69 6F  6E 73 20 6F 66 20 73 65 | eputations of se
00D760: 72 66 73 20 69 6E 20 74  68 65 69 72 20 66 75 6C | rfs in their ful
00D770: 6C 2D 73 6B 69 72 74 65  64 20 62 6C 75 65 0A 63 | l-skirted blue.c
00D780: 6F 61 74 73 2E 0A 0A 57  68 61 74 20 50 69 65 72 | oats...What Pier
00D790: 72 65 20 64 69 64 20 6E  6F 74 20 6B 6E 6F 77 20 | re did not know
00D7A0: 77 61 73 20 74 68 61 74  20 74 68 65 20 70 6C 61 | was that the pla
00D7B0: 63 65 20 77 68 65 72 65  20 74 68 65 79 20 70 72 | ce where they pr
00D7C0: 65 73 65 6E 74 65 64 20  68 69 6D 0A 77 69 74 68 | esented him.with
00D7D0: 20 62 72 65 61 64 20 61  6E 64 20 73 61 6C 74 20 |  bread and salt
00D7E0: 61 6E 64 20 77 69 73 68  65 64 20 74 6F 20 62 75 | and wished to bu
00D7F0: 69 6C 64 20 61 20 63 68  61 6E 74 00 6B 00 00 00 | ild a chant.k...
```

```
00D800: 72 79 20 69 6E 20 68 6F  6E 6F 72 20 6F 66 20 50 | ry in honor of P
00D810: 65 74 65 72 20 61 6E 64  0A 50 61 75 6C 20 77 61 | eter and.Paul wa
00D820: 73 20 61 20 6D 61 72 6B  65 74 20 76 69 6C 6C 61 | s a market villa
00D830: 67 65 20 77 68 65 72 65  20 61 20 66 61 69 72 20 | ge where a fair
00D840: 77 61 73 20 68 65 6C 64  20 6F 6E 20 53 74 2E 20 | was held on St.
00D850: 50 65 74 65 72 E2 80 99  73 20 64 61 79 2C 0A 61 | Peter’s day,.a
00D860: 6E 64 20 74 68 61 74 20  74 68 65 20 72 69 63 68 | nd that the rich
00D870: 65 73 74 20 70 65 61 73  61 6E 74 73 20 28 77 68 | est peasants (wh
00D880: 6F 20 66 6F 72 6D 65 64  20 74 68 65 20 64 65 70 | o formed the dep
00D890: 75 74 61 74 69 6F 6E 29  20 68 61 64 20 62 65 67 | utation) had beg
00D8A0: 75 6E 0A 74 68 65 20 63  68 61 6E 74 72 79 20 6C | un.the chantry l
00D8B0: 6F 6E 67 20 62 65 66 6F  72 65 2C 20 62 75 74 20 | ong before, but
00D8C0: 74 68 61 74 20 6E 69 6E  65 20 74 65 6E 74 68 73 | that nine tenths
00D8D0: 20 6F 66 20 74 68 65 20  70 65 61 73 61 6E 74 73 |  of the peasants
00D8E0: 20 69 6E 20 74 68 61 74  0A 76 69 6C 6C 61 67 65 |  in that.village
00D8F0: 73 20 77 65 72 65 20 69  6E 20 61 20 73 74 61 74 | s were in a stat
```

```
00D900: 65 20 6F 66 20 74 68 65  20 67 72 65 61 74 65 73 | e of the greates
00D910: 74 20 70 6F 76 65 72 74  79 2E 20 48 65 20 64 69 | t poverty. He di
00D920: 64 20 6E 6F 74 20 6B 6E  6F 77 20 74 68 61 74 0A | d not know that.
00D930: 73 69 6E 63 65 20 74 68  65 20 6E 75 72 73 69 6E | since the nursin
00D940: 67 20 6D 6F 74 68 65 72  73 20 77 65 72 65 20 6E | g mothers were n
00D950: 6F 20 6C 6F 6E 67 65 72  20 73 65 6E 74 20 74 6F | o longer sent to
00D960: 20 77 6F 72 6B 20 6F 6E  20 68 69 73 20 6C 61 6E |  work on his lan
00D970: 64 2C 20 74 68 65 79 0A  64 69 64 20 73 74 69 6C | d, they.did stil
00D980: 6C 20 68 61 72 64 65 72  20 77 6F 72 6B 20 6F 6E | l harder work on
00D990: 20 74 68 65 69 72 20 6F  77 6E 20 6C 61 6E 64 2E |  their own land.
00D9A0: 20 48 65 20 64 69 64 20  6E 6F 74 20 6B 6E 6F 77 |  He did not know
00D9B0: 20 74 68 61 74 20 74 68  65 20 70 72 69 65 73 74 |  that the priest
00D9C0: 0A 77 68 6F 20 6D 65 74  20 68 69 6D 20 77 69 74 | .who met him wit
00D9D0: 68 20 74 68 65 20 63 72  6F 73 73 20 6F 70 70 72 | h the cross oppr
00D9E0: 65 73 73 65 64 20 74 68  65 20 70 65 61 73 61 6E | essed the peasan
00D9F0: 74 73 20 62 79 20 68 69  73 20 65 00 6C 00 00 00 | ts by his e.l...
```

```
00DA00: 78 61 63 74 69 6F 6E 73  2C 20 61 6E 64 0A 74 68 | xactions, and.th
00DA10: 61 74 20 74 68 65 20 70  75 70 69 6C 73 E2 80 99 | at the pupils’
00DA20: 20 70 61 72 65 6E 74 73  20 77 65 70 74 20 61 74 |  parents wept at
00DA30: 20 68 61 76 69 6E 67 20  74 6F 20 6C 65 74 20 68 |  having to let h
00DA40: 69 6D 20 74 61 6B 65 20  74 68 65 69 72 20 63 68 | im take their ch
00DA50: 69 6C 64 72 65 6E 0A 61  6E 64 20 73 65 63 75 72 | ildren.and secur
00DA60: 65 64 20 74 68 65 69 72  20 72 65 6C 65 61 73 65 | ed their release
00DA70: 20 62 79 20 68 65 61 76  79 20 70 61 79 6D 65 6E |  by heavy paymen
00DA80: 74 73 2E 20 48 65 20 64  69 64 20 6E 6F 74 20 6B | ts. He did not k
00DA90: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 62 72 69 | now that the.bri
00DAA0: 63 6B 20 62 75 69 6C 64  69 6E 67 73 2C 20 62 75 | ck buildings, bu
00DAB0: 69 6C 74 20 74 6F 20 70  6C 61 6E 2C 20 77 65 72 | ilt to plan, wer
00DAC0: 65 20 62 65 69 6E 67 20  62 75 69 6C 74 20 62 79 | e being built by
00DAD0: 20 73 65 72 66 73 20 77  68 6F 73 65 20 6D 61 6E |  serfs whose man
00DAE0: 6F 72 69 61 6C 0A 6C 61  62 6F 72 20 77 61 73 20 | orial.labor was
00DAF0: 74 68 75 73 20 69 6E 63  72 65 61 73 65 64 2C 20 | thus increased,
```

```
00DB00: 74 68 6F 75 67 68 20 6C  65 73 73 65 6E 65 64 20 | though lessened
00DB10: 6F 6E 20 70 61 70 65 72  2E 20 48 65 20 64 69 64 | on paper. He did
00DB20: 20 6E 6F 74 20 6B 6E 6F  77 0A 74 68 61 74 20 77 |  not know.that w
00DB30: 68 65 72 65 20 74 68 65  20 73 74 65 77 61 72 64 | here the steward
00DB40: 20 68 61 64 20 73 68 6F  77 6E 20 68 69 6D 20 69 |  had shown him i
00DB50: 6E 20 74 68 65 20 61 63  63 6F 75 6E 74 73 20 74 | n the accounts t
00DB60: 68 61 74 20 74 68 65 20  73 65 72 66 73 E2 80 99 | hat the serfs’
00DB70: 0A 70 61 79 6D 65 6E 74  73 20 68 61 64 20 62 65 | .payments had be
00DB80: 65 6E 20 64 69 6D 69 6E  69 73 68 65 64 20 62 79 | en diminished by
00DB90: 20 61 20 74 68 69 72 64  2C 20 74 68 65 69 72 20 |  a third, their
00DBA0: 6F 62 6C 69 67 61 74 6F  72 79 20 6D 61 6E 6F 72 | obligatory manor
00DBB0: 69 61 6C 20 77 6F 72 6B  0A 68 61 64 20 62 65 65 | ial work.had bee
00DBC0: 6E 20 69 6E 63 72 65 61  73 65 64 20 62 79 20 61 | n increased by a
00DBD0: 20 68 61 6C 66 2E 20 41  6E 64 20 73 6F 20 50 69 |  half. And so Pi
00DBE0: 65 72 72 65 20 77 61 73  20 64 65 6C 69 67 68 74 | erre was delight
00DBF0: 65 64 20 77 69 74 68 20  68 69 73 00 6D 00 00 00 | ed with his.m...
```

```
00DC00: 20 76 69 73 69 74 0A 74  6F 20 68 69 73 20 65 73 |  visit.to his es
00DC10: 74 61 74 65 73 20 61 6E  64 20 71 75 69 74 65 20 | tates and quite
00DC20: 72 65 63 6F 76 65 72 65  64 20 74 68 65 20 70 68 | recovered the ph
00DC30: 69 6C 61 6E 74 68 72 6F  70 69 63 20 6D 6F 6F 64 | ilanthropic mood
00DC40: 20 69 6E 20 77 68 69 63  68 0A 68 65 20 68 61 64 |  in which.he had
00DC50: 20 6C 65 66 74 20 50 65  74 65 72 73 62 75 72 67 |  left Petersburg
00DC60: 2C 20 61 6E 64 20 77 72  6F 74 65 20 65 6E 74 68 | , and wrote enth
00DC70: 75 73 69 61 73 74 69 63  20 6C 65 74 74 65 72 73 | usiastic letters
00DC80: 20 74 6F 20 68 69 73 0A  E2 80 9C 62 72 6F 74 68 |  to his.“broth
00DC90: 65 72 2D 69 6E 73 74 72  75 63 74 6F 72 E2 80 9D | er-instructor”
00DCA0: 20 61 73 20 68 65 20 63  61 6C 6C 65 64 20 74 68 |  as he called th
00DCB0: 65 20 47 72 61 6E 64 20  4D 61 73 74 65 72 2E 0A | e Grand Master..
00DCC0: 0A E2 80 9C 48 6F 77 20  65 61 73 79 20 69 74 20 | .“How easy it
00DCD0: 69 73 2C 20 68 6F 77 20  6C 69 74 74 6C 65 20 65 | is, how little e
00DCE0: 66 66 6F 72 74 20 69 74  20 6E 65 65 64 73 2C 20 | ffort it needs,
00DCF0: 74 6F 20 64 6F 20 73 6F  20 6D 75 63 68 20 67 6F | to do so much go
```

```
00DD00: 6F 64 2C E2 80 9D 0A 74  68 6F 75 67 68 74 20 50 | od,”.thought P
00DD10: 69 65 72 72 65 2C 20 E2  80 9C 61 6E 64 20 68 6F | ierre, “and ho
00DD20: 77 20 6C 69 74 74 6C 65  20 61 74 74 65 6E 74 69 | w little attenti
00DD30: 6F 6E 20 77 65 20 70 61  79 20 74 6F 20 69 74 21 | on we pay to it!
00DD40: E2 80 9D 0A 0A 48 65 20  77 61 73 20 70 6C 65 61 | ”..He was plea
00DD50: 73 65 64 20 61 74 20 74  68 65 20 67 72 61 74 69 | sed at the grati
00DD60: 74 75 64 65 20 68 65 20  72 65 63 65 69 76 65 64 | tude he received
00DD70: 2C 20 62 75 74 20 66 65  6C 74 20 61 62 61 73 68 | , but felt abash
00DD80: 65 64 20 61 74 0A 72 65  63 65 69 76 69 6E 67 20 | ed at.receiving
00DD90: 69 74 2E 20 54 68 69 73  20 67 72 61 74 69 74 75 | it. This gratitu
00DDA0: 64 65 20 72 65 6D 69 6E  64 65 64 20 68 69 6D 20 | de reminded him
00DDB0: 6F 66 20 68 6F 77 20 6D  75 63 68 20 6D 6F 72 65 | of how much more
00DDC0: 20 68 65 20 6D 69 67 68  74 20 64 6F 0A 66 6F 72 |  he might do.for
00DDD0: 20 74 68 65 73 65 20 73  69 6D 70 6C 65 2C 20 6B |  these simple, k
00DDE0: 69 6E 64 6C 79 20 70 65  6F 70 6C 65 2E 0A 0A 54 | indly people...T
00DDF0: 68 65 20 63 68 69 65 66  20 73 74 00 6E 00 00 00 | he chief st.n...
```

```
00DE00: 65 77 61 72 64 2C 20 61  20 76 65 72 79 20 73 74 | eward, a very st
00DE10: 75 70 69 64 20 62 75 74  20 63 75 6E 6E 69 6E 67 | upid but cunning
00DE20: 20 6D 61 6E 20 77 68 6F  20 73 61 77 20 70 65 72 |  man who saw per
00DE30: 66 65 63 74 6C 79 0A 74  68 72 6F 75 67 68 20 74 | fectly.through t
00DE40: 68 65 20 6E 61 C3 AF 76  65 20 61 6E 64 20 69 6E | he naïve and in
00DE50: 74 65 6C 6C 69 67 65 6E  74 20 63 6F 75 6E 74 20 | telligent count
00DE60: 61 6E 64 20 70 6C 61 79  65 64 20 77 69 74 68 20 | and played with
00DE70: 68 69 6D 20 61 73 20 77  69 74 68 0A 61 20 74 6F | him as with.a to
00DE80: 79 2C 20 73 65 65 69 6E  67 20 74 68 65 20 65 66 | y, seeing the ef
00DE90: 66 65 63 74 20 74 68 65  73 65 20 70 72 65 61 72 | fect these prear
00DEA0: 72 61 6E 67 65 64 20 72  65 63 65 70 74 69 6F 6E | ranged reception
00DEB0: 73 20 68 61 64 20 6F 6E  20 50 69 65 72 72 65 2C | s had on Pierre,
00DEC0: 0A 70 72 65 73 73 65 64  20 68 69 6D 20 73 74 69 | .pressed him sti
00DED0: 6C 6C 20 68 61 72 64 65  72 20 77 69 74 68 20 70 | ll harder with p
00DEE0: 72 6F 6F 66 73 20 6F 66  20 74 68 65 20 69 6D 70 | roofs of the imp
00DEF0: 6F 73 73 69 62 69 6C 69  74 79 20 61 6E 64 20 61 | ossibility and a
```

```
00DF00: 62 6F 76 65 20 61 6C 6C  0A 74 68 65 20 75 73 65 | bove all.the use
00DF10: 6C 65 73 73 6E 65 73 73  20 6F 66 20 66 72 65 65 | lessness of free
00DF20: 69 6E 67 20 74 68 65 20  73 65 72 66 73 2C 20 77 | ing the serfs, w
00DF30: 68 6F 20 77 65 72 65 20  71 75 69 74 65 20 68 61 | ho were quite ha
00DF40: 70 70 79 20 61 73 20 69  74 20 77 61 73 2E 0A 0A | ppy as it was...
00DF50: 50 69 65 72 72 65 20 69  6E 20 68 69 73 20 73 65 | Pierre in his se
00DF60: 63 72 65 74 20 73 6F 75  6C 20 61 67 72 65 65 64 | cret soul agreed
00DF70: 20 77 69 74 68 20 74 68  65 20 73 74 65 77 61 72 |  with the stewar
00DF80: 64 20 74 68 61 74 20 69  74 20 77 6F 75 6C 64 20 | d that it would
00DF90: 62 65 0A 64 69 66 66 69  63 75 6C 74 20 74 6F 20 | be.difficult to
00DFA0: 69 6D 61 67 69 6E 65 20  68 61 70 70 69 65 72 20 | imagine happier
00DFB0: 70 65 6F 70 6C 65 2C 20  61 6E 64 20 74 68 61 74 | people, and that
00DFC0: 20 47 6F 64 20 6F 6E 6C  79 20 6B 6E 65 77 20 77 |  God only knew w
00DFD0: 68 61 74 20 77 6F 75 6C  64 0A 68 61 70 70 65 6E | hat would.happen
00DFE0: 20 74 6F 20 74 68 65 6D  20 77 68 65 6E 20 74 68 |  to them when th
00DFF0: 65 79 20 77 65 72 65 20  66 72 65 00 6F 00 00 00 | ey were fre.o...
```

```
00E000: 65 2C 20 62 75 74 20 68  65 20 69 6E 73 69 73 74 | e, but he insist
00E010: 65 64 2C 20 74 68 6F 75  67 68 20 72 65 6C 75 63 | ed, though reluc
00E020: 74 61 6E 74 6C 79 2C 0A  6F 6E 20 77 68 61 74 20 | tantly,.on what
00E030: 68 65 20 74 68 6F 75 67  68 74 20 72 69 67 68 74 | he thought right
00E040: 2E 20 54 68 65 20 73 74  65 77 61 72 64 20 70 72 | . The steward pr
00E050: 6F 6D 69 73 65 64 20 74  6F 20 64 6F 20 61 6C 6C | omised to do all
00E060: 20 69 6E 20 68 69 73 20  70 6F 77 65 72 20 74 6F |  in his power to
00E070: 0A 63 61 72 72 79 20 6F  75 74 20 74 68 65 20 63 | .carry out the c
00E080: 6F 75 6E 74 E2 80 99 73  20 77 69 73 68 65 73 2C | ount’s wishes,
00E090: 20 73 65 65 69 6E 67 20  63 6C 65 61 72 6C 79 20 |  seeing clearly
00E0A0: 74 68 61 74 20 6E 6F 74  20 6F 6E 6C 79 20 77 6F | that not only wo
00E0B0: 75 6C 64 20 74 68 65 0A  63 6F 75 6E 74 20 6E 65 | uld the.count ne
00E0C0: 76 65 72 20 62 65 20 61  62 6C 65 20 74 6F 20 66 | ver be able to f
00E0D0: 69 6E 64 20 6F 75 74 20  77 68 65 74 68 65 72 20 | ind out whether
00E0E0: 61 6C 6C 20 6D 65 61 73  75 72 65 73 20 68 61 64 | all measures had
00E0F0: 20 62 65 65 6E 20 74 61  6B 65 6E 20 66 6F 72 0A |  been taken for.
```

```
00E100: 74 68 65 20 73 61 6C 65  20 6F 66 20 74 68 65 20 | the sale of the
00E110: 6C 61 6E 64 20 61 6E 64  20 66 6F 72 65 73 74 73 | land and forests
00E120: 20 61 6E 64 20 74 6F 20  72 65 6C 65 61 73 65 20 |  and to release
00E130: 74 68 65 6D 20 66 72 6F  6D 20 74 68 65 20 4C 61 | them from the La
00E140: 6E 64 20 42 61 6E 6B 2C  0A 62 75 74 20 77 6F 75 | nd Bank,.but wou
00E150: 6C 64 20 70 72 6F 62 61  62 6C 79 20 6E 65 76 65 | ld probably neve
00E160: 72 20 65 76 65 6E 20 69  6E 71 75 69 72 65 20 61 | r even inquire a
00E170: 6E 64 20 77 6F 75 6C 64  20 6E 65 76 65 72 20 6B | nd would never k
00E180: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 6E 65 77 | now that the.new
00E190: 6C 79 20 65 72 65 63 74  65 64 20 62 75 69 6C 64 | ly erected build
00E1A0: 69 6E 67 73 20 77 65 72  65 20 73 74 61 6E 64 69 | ings were standi
00E1B0: 6E 67 20 65 6D 70 74 79  20 61 6E 64 20 74 68 61 | ng empty and tha
00E1C0: 74 20 74 68 65 20 73 65  72 66 73 20 63 6F 6E 74 | t the serfs cont
00E1D0: 69 6E 75 65 64 0A 74 6F  20 67 69 76 65 20 69 6E | inued.to give in
00E1E0: 20 6D 6F 6E 65 79 20 61  6E 64 20 77 6F 72 6B 20 |  money and work
00E1F0: 61 6C 6C 20 74 68 61 74  20 6F 74 00 70 00 00 00 | all that ot.p...
```

```
00E200: 68 65 72 20 70 65 6F 70  6C 65 E2 80 99 73 20 73 | her people’s s
00E210: 65 72 66 73 20 67 61 76  65 E2 80 94 74 68 61 74 | erfs gave—that
00E220: 20 69 73 0A 74 6F 20 73  61 79 2C 20 61 6C 6C 20 |  is.to say, all
00E230: 74 68 61 74 20 63 6F 75  6C 64 20 62 65 20 67 6F | that could be go
00E240: 74 20 6F 75 74 20 6F 66  20 74 68 65 6D 2E 70 72 | t out of them.pr
00E250: 6F 6D 69 73 65 64 20 74  6F 20 64 6F 20 61 6C 6C | omised to do all
00E260: 20 69 6E 20 68 69 73 20  70 6F 77 65 72 20 74 6F |  in his power to
00E270: 0A 63 61 72 72 79 20 6F  75 74 20 74 68 65 20 63 | .carry out the c
00E280: 6F 75 6E 74 E2 80 99 73  20 77 69 73 68 65 73 2C | ount’s wishes,
00E290: 20 73 65 65 69 6E 67 20  63 6C 65 61 72 6C 79 20 |  seeing clearly
00E2A0: 74 68 61 74 20 6E 6F 74  20 6F 6E 6C 79 20 77 6F | that not only wo
00E2B0: 75 6C 64 20 74 68 65 0A  63 6F 75 6E 74 20 6E 65 | uld the.count ne
00E2C0: 76 65 72 20 62 65 20 61  62 6C 65 20 74 6F 20 66 | ver be able to f
00E2D0: 69 6E 64 20 6F 75 74 20  77 68 65 74 68 65 72 20 | ind out whether
00E2E0: 61 6C 6C 20 6D 65 61 73  75 72 65 73 20 68 61 64 | all measures had
00E2F0: 20 62 65 65 6E 20 74 61  6B 65 6E 20 66 6F 72 0A |  been taken for.
```

```
00E300: 74 68 65 20 73 61 6C 65  20 6F 66 20 74 68 65 20 | the sale of the
00E310: 6C 61 6E 64 20 61 6E 64  20 66 6F 72 65 73 74 73 | land and forests
00E320: 20 61 6E 64 20 74 6F 20  72 65 6C 65 61 73 65 20 |  and to release
00E330: 74 68 65 6D 20 66 72 6F  6D 20 74 68 65 20 4C 61 | them from the La
00E340: 6E 64 20 42 61 6E 6B 2C  0A 62 75 74 20 77 6F 75 | nd Bank,.but wou
00E350: 6C 64 20 70 72 6F 62 61  62 6C 79 20 6E 65 76 65 | ld probably neve
00E360: 72 20 65 76 65 6E 20 69  6E 71 75 69 72 65 20 61 | r even inquire a
00E370: 6E 64 20 77 6F 75 6C 64  20 6E 65 76 65 72 20 6B | nd would never k
00E380: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 6E 65 77 | now that the.new
00E390: 6C 79 20 65 72 65 63 74  65 64 20 62 75 69 6C 64 | ly erected build
00E3A0: 69 6E 67 73 20 77 65 72  65 20 73 74 61 6E 64 69 | ings were standi
00E3B0: 6E 67 20 65 6D 70 74 79  20 61 6E 64 20 74 68 61 | ng empty and tha
00E3C0: 74 20 74 68 65 20 73 65  72 66 73 20 63 6F 6E 74 | t the serfs cont
00E3D0: 69 6E 75 65 64 0A 74 6F  20 67 69 76 65 20 69 6E | inued.to give in
00E3E0: 20 6D 6F 6E 65 79 20 61  6E 64 20 77 6F 72 6B 20 |  money and work
00E3F0: 61 6C 6C 20 74 68 61 74  20 6F 74 00 70 00 00 00 | all that ot.p...
```

cp2l

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > cp2l test_copy.txt c.txt
Prompt > exit
System exiting
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

cat for cp2l

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ cat test.txt
The southern spring, the comfortable rapid traveling in a Vienna
carriage, and the solitude of the road, all had a gladdening effect on
Pierre. The estates he had not before visited were each more picturesque
than the other; the serfs everywhere seemed thriving and touchingly
grateful for the benefits conferred on them. Everywhere were receptions,
which though they embarrassed Pierre awakened a joyful feeling in the
depth of his heart. In one place the peasants presented him with bread
and salt and an icon of Saint Peter and Saint Paul, asking permission,
as a mark of their gratitude for the benefits he had conferred on them,
to build a new chantry to the church at their own expense in honor
of Peter and Paul, his patron saints. In another place the women with
infants in arms met him to thank him for releasing them from hard
work. On a third estate the priest, bearing a cross, came to meet
him surrounded by children whom, by the count’s generosity, he was
instructing in reading, writing, and religion. On all his estates Pierre
saw with his own eyes brick buildings erected or in course of erection,
all on one plan, for hospitals, schools, and almshouses, which were soon
to be opened. Everywhere he saw the stewards’ accounts, according to
which the serfs’ manorial labor had been diminished, and heard the
touching thanks of deputations of serfs in their full-skirted blue
coats.
```

```
What Pierre did not know was that the place where they presented him
with bread and salt and wished to build a chantry in honor of Peter and
Paul was a market village where a fair was held on St. Peter's day,
and that the richest peasants (who formed the deputation) had begun
the chantry long before, but that nine tenths of the peasants in that
villages were in a state of the greatest poverty. He did not know that
since the nursing mothers were no longer sent to work on his land, they
did still harder work on their own land. He did not know that the priest
who met him with the cross oppressed the peasants by his exactions, and
that the pupils' parents wept at having to let him take their children
and secured their release by heavy payments. He did not know that the
brick buildings, built to plan, were being built by serfs whose manorial
labor was thus increased, though lessened on paper. He did not know
that where the steward had shown him in the accounts that the serfs'
payments had been diminished by a third, their obligatory manorial work
had been increased by a half. And so Pierre was delighted with his visit
to his estates and quite recovered the philanthropic mood in which
he had left Petersburg, and wrote enthusiastic letters to his
"brother-instructor" as he called the Grand Master.

"How easy it is, how little effort it needs, to do so much good,"
thought Pierre, "and how little attention we pay to it!"

He was pleased at the gratitude he received, but felt abashed at
receiving it. This gratitude reminded him of how much more he might do
for these simple, kindly people.
```

```
The chief steward, a very stupid but cunning man who saw perfectly
through the naïve and intelligent count and played with him as with
a toy, seeing the effect these prearranged receptions had on Pierre,
pressed him still harder with proofs of the impossibility and above all
the uselessness of freeing the serfs, who were quite happy as it was.

Pierre in his secret soul agreed with the steward that it would be
difficult to imagine happier people, and that God only knew what would
happen to them when they were free, but he insisted, though reluctantly,
on what he thought right. The steward promised to do all in his power to
carry out the count's wishes, seeing clearly that not only would the
count never be able to find out whether all measures had been taken for
the sale of the land and forests and to release them from the Land Bank,
but would probably never even inquire and would never know that the
newly erected buildings were standing empty and that the serfs continued
to give in money and work all that other people's serfs gave—that is
to say, all that could be got out of them.student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ cat c.txt
The southern spring, the comfortable rapid traveling in a Vienna
carriage, and the solitude of the road, all had a gladdening effect on
Pierre. The estates he had not before visited were each more picturesque
than the other; the serfs everywhere seemed thriving and touchingly
grateful for the benefits conferred on them. Everywhere were receptions,
which though they embarrassed Pierre awakened a joyful feeling in the
depth of his heart. In one place the peasants presented him with bread
and salt and an icon of Saint Peter and Saint Paul, asking permission,
as a mark of their gratitude for the benefits he had conferred on them,
to build a new chantry to the church at their own expense in honor
of Peter and Paul, his patron saints. In another place the women with
infants in arms met him to thank him for releasing them from hard
work. On a third estate the priest, bearing a cross, came to meet
him surrounded by children whom, by the count's generosity, he was
instructing in reading, writing, and religion. On all his estates Pierre
saw with his own eyes brick buildings erected or in course of erection,
all on one plan, for hospitals, schools, and almshouses, which were soon
to be opened. Everywhere he saw the stewards' accounts, according to
which the serfs' manorial labor had been diminished, and heard the
touching thanks of deputations of serfs in their full-skirted blue
coats.
```

```
What Pierre did not know was that the place where they presented him
with bread and salt and wished to build a chantry in honor of Peter and
Paul was a market village where a fair was held on St. Peter's day,
and that the richest peasants (who formed the deputation) had begun
the chantry long before, but that nine tenths of the peasants in that
villages were in a state of the greatest poverty. He did not know that
since the nursing mothers were no longer sent to work on his land, they
did still harder work on their own land. He did not know that the priest
who met him with the cross oppressed the peasants by his exactions, and
that the pupils' parents wept at having to let him take their children
and secured their release by heavy payments. He did not know that the
brick buildings, built to plan, were being built by serfs whose manorial
labor was thus increased, though lessened on paper. He did not know
that where the steward had shown him in the accounts that the serfs'
payments had been diminished by a third, their obligatory manorial work
had been increased by a half. And so Pierre was delighted with his visit
to his estates and quite recovered the philanthropic mood in which
he had left Petersburg, and wrote enthusiastic letters to his
"brother-instructor" as he called the Grand Master.

"How easy it is, how little effort it needs, to do so much good,"
thought Pierre, "and how little attention we pay to it!"

He was pleased at the gratitude he received, but felt abashed at
receiving it. This gratitude reminded him of how much more he might do
for these simple, kindly people.

The chief steward, a very stupid but cunning man who saw perfectly
through the naïve and intelligent count and played with him as with
a toy, seeing the effect these prearranged receptions had on Pierre,
pressed him still harder with proofs of the impossibility and above all
the uselessness of freeing the serfs, who were quite happy as it was.

Pierre in his secret soul agreed with the steward that it would be
difficult to imagine happier people, and that God only knew what would
happen to them when they were free, but he insisted, though reluctantly,
on what he thought right. The steward promised to do all in his power to
carry out the count's wishes, seeing clearly that not only would the
count never be able to find out whether all measures had been taken for
the sale of the land and forests and to release them from the Land Bank,
but would probably never even inquire and would never know that the
newly erected buildings were standing empty and that the serfs continued
to give in money and work all that other people's serfs gave—that is
to say, all that could be got out of them.student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$
```

cp

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ make run
./fsshell SampleVolume 10000000 512
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
Opened SampleVolume, Volume Size: 9999872;  BlockSize: 512; Return 0
Initializing File System with 19531 blocks with a block size of 512
Prompt > cp c.txt d.txt
Prompt > exit
System exiting
```

hexdump for cp

```
student@student-VirtualBox:~/csc415-filesystem-BeeSeeWhy$ Hexdump/hexdump.linux
--count 9 --start 123 SampleVolume
Dumping file SampleVolume, starting at block 123 for 9 blocks:

00F600: 54 68 65 20 73 6F 75 74  68 65 72 6E 20 73 70 72 | The southern spr
00F610: 69 6E 67 2C 20 74 68 65  20 63 6F 6D 66 6F 72 74 | ing, the comfort
00F620: 61 62 6C 65 20 72 61 70  69 64 20 74 72 61 76 65 | able rapid trave
00F630: 6C 69 6E 67 20 69 6E 20  61 20 56 69 65 6E 6E 61 | ling in a Vienna
00F640: 0A 63 61 72 72 69 61 67  65 2C 20 61 6E 64 20 74 | .carriage, and t
00F650: 68 65 20 73 6F 6C 69 74  75 64 65 20 6F 66 20 74 | he solitude of t
00F660: 68 65 20 72 6F 61 64 2C  20 61 6C 6C 20 68 61 64 | he road, all had
00F670: 20 61 20 67 6C 61 64 64  65 6E 69 6E 67 20 65 66 |  a gladdening ef
00F680: 66 65 63 74 20 6F 6E 0A  50 69 65 72 72 65 2E 20 | fect on.Pierre.
00F690: 54 68 65 20 65 73 74 61  74 65 73 20 68 65 20 68 | The estates he h
00F6A0: 61 64 20 6E 6F 74 20 62  65 66 6F 72 65 20 76 69 | ad not before vi
00F6B0: 73 69 74 65 64 20 77 65  72 65 20 65 61 63 68 20 | sited were each
00F6C0: 6D 6F 72 65 20 70 69 63  74 75 72 65 73 71 75 65 | more picturesque
00F6D0: 0A 74 68 61 6E 20 74 68  65 20 6F 74 68 65 72 3B | .than the other;
00F6E0: 20 74 68 65 20 73 65 72  66 73 20 65 76 65 72 79 |  the serfs every
00F6F0: 77 68 65 72 65 20 73 65  65 6D 65 64 20 74 68 72 | where seemed thr
```

```
00F700: 69 76 69 6E 67 20 61 6E  64 20 74 6F 75 63 68 69 | iving and touchi
00F710: 6E 67 6C 79 0A 67 72 61  74 65 66 75 6C 20 66 6F | ngly.grateful fo
00F720: 72 20 74 68 65 20 62 65  6E 65 66 69 74 73 20 63 | r the benefits c
00F730: 6F 6E 66 65 72 72 65 64  20 6F 6E 20 74 68 65 6D | onferred on them
00F740: 2E 20 45 76 65 72 79 77  68 65 72 65 20 77 65 72 | . Everywhere wer
00F750: 65 20 72 65 63 65 70 74  69 6F 6E 73 2C 0A 77 68 | e receptions,.wh
00F760: 69 63 68 20 74 68 6F 75  67 68 20 74 68 65 79 20 | ich though they
00F770: 65 6D 62 61 72 72 61 73  73 65 64 20 50 69 65 72 | embarrassed Pier
00F780: 72 65 20 61 77 61 6B 65  6E 65 64 20 61 20 6A 6F | re awakened a jo
00F790: 79 66 75 6C 20 66 65 65  6C 69 6E 67 20 69 6E 20 | yful feeling in
00F7A0: 74 68 65 0A 64 65 70 74  68 20 6F 66 20 68 69 73 | the.depth of his
00F7B0: 20 68 65 61 72 74 2E 20  49 6E 20 6F 6E 65 20 70 |  heart. In one p
00F7C0: 6C 61 63 65 20 74 68 65  20 70 65 61 73 61 6E 74 | lace the peasant
00F7D0: 73 20 70 72 65 73 65 6E  74 65 64 20 68 69 6D 20 | s presented him
00F7E0: 77 69 74 68 20 62 72 65  61 64 0A 61 6E 64 20 73 | with bread.and s
00F7F0: 61 6C 74 20 61 6E 64 20  61 6E 20 00 7B 00 00 00 | alt and an .{...
```


```
00F700: 69 76 69 6E 67 20 61 6E  64 20 74 6F 75 63 68 69 | iving and touchi
00F710: 6E 67 6C 79 0A 67 72 61  74 65 66 75 6C 20 66 6F | ngly.grateful fo
00F720: 72 20 74 68 65 20 62 65  6E 65 66 69 74 73 20 63 | r the benefits c
00F730: 6F 6E 66 65 72 72 65 64  20 6F 6E 20 74 68 65 6D | onferred on them
00F740: 2E 20 45 76 65 72 79 77  68 65 72 65 20 77 65 72 | . Everywhere wer
00F750: 65 20 72 65 63 65 70 74  69 6F 6E 73 2C 0A 77 68 | e receptions,.wh
00F760: 69 63 68 20 74 68 6F 75  67 68 20 74 68 65 79 20 | ich though they
00F770: 65 6D 62 61 72 72 61 73  73 65 64 20 50 69 65 72 | embarrassed Pier
00F780: 72 65 20 61 77 61 6B 65  6E 65 64 20 61 20 6A 6F | re awakened a jo
00F790: 79 66 75 6C 20 66 65 65  6C 69 6E 67 20 69 6E 20 | yful feeling in
00F7A0: 74 68 65 0A 64 65 70 74  68 20 6F 66 20 68 69 73 | the.depth of his
00F7B0: 20 68 65 61 72 74 2E 20  49 6E 20 6F 6E 65 20 70 |  heart. In one p
00F7C0: 6C 61 63 65 20 74 68 65  20 70 65 61 73 61 6E 74 | lace the peasant
00F7D0: 73 20 70 72 65 73 65 6E  74 65 64 20 68 69 6D 20 | s presented him
00F7E0: 77 69 74 68 20 62 72 65  61 64 0A 61 6E 64 20 73 | with bread.and s
00F7F0: 61 6C 74 20 61 6E 64 20  61 6E 20 00 7B 00 00 00 | alt and an .{...
```


```
00F900: 65 6E 20 77 69 74 68 0A  69 6E 66 61 6E 74 73 20 | en with.infants
00F910: 69 6E 20 61 72 6D 73 20  6D 65 74 20 68 69 6D 20 | in arms met him
00F920: 74 6F 20 74 68 61 6E 6B  20 68 69 6D 20 66 6F 72 | to thank him for
00F930: 20 72 65 6C 65 61 73 69  6E 67 20 74 68 65 6D 20 |  releasing them
00F940: 66 72 6F 6D 20 68 61 72  64 0A 77 6F 72 6B 2E 20 | from hard.work.
00F950: 4F 6E 20 61 20 74 68 69  72 64 20 65 73 74 61 74 | On a third estat
00F960: 65 20 74 68 65 20 70 72  69 65 73 74 2C 20 62 65 | e the priest, be
00F970: 61 72 69 6E 67 20 61 20  63 72 6F 73 73 2C 20 63 | aring a cross, c
00F980: 61 6D 65 20 74 6F 20 6D  65 65 74 0A 68 69 6D 20 | ame to meet.him
00F990: 73 75 72 72 6F 75 6E 64  65 64 20 62 79 20 63 68 | surrounded by ch
00F9A0: 69 6C 64 72 65 6E 20 77  68 6F 6D 2C 20 62 79 20 | ildren whom, by
00F9B0: 74 68 65 20 63 6F 75 6E  74 E2 80 99 73 20 67 65 | the count’s ge
00F9C0: 6E 65 72 6F 73 69 74 79  2C 20 68 65 20 77 61 73 | nerosity, he was
00F9D0: 0A 69 6E 73 74 72 75 63  74 69 6E 67 20 69 6E 20 | .instructing in
00F9E0: 72 65 61 64 69 6E 67 2C  20 77 72 69 74 69 6E 67 | reading, writing
00F9F0: 2C 20 61 6E 64 20 72 65  6C 69 67 00 7C 00 00 00 | , and relig.|...
```

```
00FA00: 69 6F 6E 2E 20 4F 6E 20  61 6C 6C 20 68 69 73 20 | ion. On all his
00FA10: 65 73 74 61 74 65 73 20  50 69 65 72 72 65 0A 73 | estates Pierre.s
00FA20: 61 77 20 77 69 74 68 20  68 69 73 20 6F 77 6E 20 | aw with his own
00FA30: 65 79 65 73 20 62 72 69  63 6B 20 62 75 69 6C 64 | eyes brick build
00FA40: 69 6E 67 73 20 65 72 65  63 74 65 64 20 6F 72 20 | ings erected or
00FA50: 69 6E 20 63 6F 75 72 73  65 20 6F 66 20 65 72 65 | in course of ere
00FA60: 63 74 69 6F 6E 2C 0A 61  6C 6C 20 6F 6E 20 6F 6E | ction,.all on on
00FA70: 65 20 70 6C 61 6E 2C 20  66 6F 72 20 68 6F 73 70 | e plan, for hosp
00FA80: 69 74 61 6C 73 2C 20 73  63 68 6F 6F 6C 73 2C 20 | itals, schools,
00FA90: 61 6E 64 20 61 6C 6D 73  68 6F 75 73 65 73 2C 20 | and almshouses,
00FAA0: 77 68 69 63 68 20 77 65  72 65 20 73 6F 6F 6E 0A | which were soon.
00FAB0: 74 6F 20 62 65 20 6F 70  65 6E 65 64 2E 20 45 76 | to be opened. Ev
00FAC0: 65 72 79 77 68 65 72 65  20 68 65 20 73 61 77 20 | erywhere he saw
00FAD0: 74 68 65 20 73 74 65 77  61 72 64 73 E2 80 99 20 | the stewards’
00FAE0: 61 63 63 6F 75 6E 74 73  2C 20 61 63 63 6F 72 64 | accounts, accord
00FAF0: 69 6E 67 20 74 6F 0A 77  68 69 63 68 20 74 68 65 | ing to.which the
```

```
00FB00: 20 73 65 72 66 73 E2 80  99 20 6D 61 6E 6F 72 69 |  serfs’ manori
00FB10: 61 6C 20 6C 61 62 6F 72  20 68 61 64 20 62 65 65 | al labor had bee
00FB20: 6E 20 64 69 6D 69 6E 69  73 68 65 64 2C 20 61 6E | n diminished, an
00FB30: 64 20 68 65 61 72 64 20  74 68 65 0A 74 6F 75 63 | d heard the.touc
00FB40: 68 69 6E 67 20 74 68 61  6E 6B 73 20 6F 66 20 64 | hing thanks of d
00FB50: 65 70 75 74 61 74 69 6F  6E 73 20 6F 66 20 73 65 | eputations of se
00FB60: 72 66 73 20 69 6E 20 74  68 65 69 72 20 66 75 6C | rfs in their ful
00FB70: 6C 2D 73 6B 69 72 74 65  64 20 62 6C 75 65 0A 63 | l-skirted blue.c
00FB80: 6F 61 74 73 2E 0A 0A 57  68 61 74 20 50 69 65 72 | oats...What Pier
00FB90: 72 65 20 64 69 64 20 6E  6F 74 20 6B 6E 6F 77 20 | re did not know
00FBA0: 77 61 73 20 74 68 61 74  20 74 68 65 20 70 6C 61 | was that the pla
00FBB0: 63 65 20 77 68 65 72 65  20 74 68 65 79 20 70 72 | ce where they pr
00FBC0: 65 73 65 6E 74 65 64 20  68 69 6D 0A 77 69 74 68 | esented him.with
00FBD0: 20 62 72 65 61 64 20 61  6E 64 20 73 61 6C 74 20 |  bread and salt
00FBE0: 61 6E 64 20 77 69 73 68  65 64 20 74 6F 20 62 75 | and wished to bu
00FBF0: 69 6C 64 20 61 20 63 68  61 6E 74 00 7D 00 00 00 | ild a chant.}...
```

```
00FC00: 72 79 20 69 6E 20 68 6F  6E 6F 72 20 6F 66 20 50 | ry in honor of P
00FC10: 65 74 65 72 20 61 6E 64  0A 50 61 75 6C 20 77 61 | eter and.Paul wa
00FC20: 73 20 61 20 6D 61 72 6B  65 74 20 76 69 6C 6C 61 | s a market villa
00FC30: 67 65 20 77 68 65 72 65  20 61 20 66 61 69 72 20 | ge where a fair
00FC40: 77 61 73 20 68 65 6C 64  20 6F 6E 20 53 74 2E 20 | was held on St.
00FC50: 50 65 74 65 72 E2 80 99  73 20 64 61 79 2C 0A 61 | Peter’s day,.a
00FC60: 6E 64 20 74 68 61 74 20  74 68 65 20 72 69 63 68 | nd that the rich
00FC70: 65 73 74 20 70 65 61 73  61 6E 74 73 20 28 77 68 | est peasants (wh
00FC80: 6F 20 66 6F 72 6D 65 64  20 74 68 65 20 64 65 70 | o formed the dep
00FC90: 75 74 61 74 69 6F 6E 29  20 68 61 64 20 62 65 67 | utation) had beg
00FCA0: 75 6E 0A 74 68 65 20 63  68 61 6E 74 72 79 20 6C | un.the chantry l
00FCB0: 6F 6E 67 20 62 65 66 6F  72 65 2C 20 62 75 74 20 | ong before, but
00FCC0: 74 68 61 74 20 6E 69 6E  65 20 74 65 6E 74 68 73 | that nine tenths
00FCD0: 20 6F 66 20 74 68 65 20  70 65 61 73 61 6E 74 73 |  of the peasants
00FCE0: 20 69 6E 20 74 68 61 74  0A 76 69 6C 6C 61 67 65 |  in that.village
00FCF0: 73 20 77 65 72 65 20 69  6E 20 61 20 73 74 61 74 | s were in a stat
```

```
00FD00: 65 20 6F 66 20 74 68 65  20 67 72 65 61 74 65 73 | e of the greates
00FD10: 74 20 70 6F 76 65 72 74  79 2E 20 48 65 20 64 69 | t poverty. He di
00FD20: 64 20 6E 6F 74 20 6B 6E  6F 77 20 74 68 61 74 0A | d not know that.
00FD30: 73 69 6E 63 65 20 74 68  65 20 6E 75 72 73 69 6E | since the nursin
00FD40: 67 20 6D 6F 74 68 65 72  73 20 77 65 72 65 20 6E | g mothers were n
00FD50: 6F 20 6C 6F 6E 67 65 72  20 73 65 6E 74 20 74 6F | o longer sent to
00FD60: 20 77 6F 72 6B 20 6F 6E  20 68 69 73 20 6C 61 6E |  work on his lan
00FD70: 64 2C 20 74 68 65 79 0A  64 69 64 20 73 74 69 6C | d, they.did stil
00FD80: 6C 20 68 61 72 64 65 72  20 77 6F 72 6B 20 6F 6E | l harder work on
00FD90: 20 74 68 65 69 72 20 6F  77 6E 20 6C 61 6E 64 2E |  their own land.
00FDA0: 20 48 65 20 64 69 64 20  6E 6F 74 20 6B 6E 6F 77 |  He did not know
00FDB0: 20 74 68 61 74 20 74 68  65 20 70 72 69 65 73 74 |  that the priest
00FDC0: 0A 77 68 6F 20 6D 65 74  20 68 69 6D 20 77 69 74 | .who met him wit
00FDD0: 68 20 74 68 65 20 63 72  6F 73 73 20 6F 70 70 72 | h the cross oppr
00FDE0: 65 73 73 65 64 20 74 68  65 20 70 65 61 73 61 6E | essed the peasan
00FDF0: 74 73 20 62 79 20 68 69  73 20 65 00 7E 00 00 00 | ts by his e.~...
```

```
00FE00: 78 61 63 74 69 6F 6E 73  2C 20 61 6E 64 0A 74 68 | xactions, and.th
00FE10: 61 74 20 74 68 65 20 70  75 70 69 6C 73 E2 80 99 | at the pupils’
00FE20: 20 70 61 72 65 6E 74 73  20 77 65 70 74 20 61 74 |  parents wept at
00FE30: 20 68 61 76 69 6E 67 20  74 6F 20 6C 65 74 20 68 |  having to let h
00FE40: 69 6D 20 74 61 6B 65 20  74 68 65 69 72 20 63 68 | im take their ch
00FE50: 69 6C 64 72 65 6E 0A 61  6E 64 20 73 65 63 75 72 | ildren.and secur
00FE60: 65 64 20 74 68 65 69 72  20 72 65 6C 65 61 73 65 | ed their release
00FE70: 20 62 79 20 68 65 61 76  79 20 70 61 79 6D 65 6E |  by heavy paymen
00FE80: 74 73 2E 20 48 65 20 64  69 64 20 6E 6F 74 20 6B | ts. He did not k
00FE90: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 62 72 69 | now that the.bri
00FEA0: 63 6B 20 62 75 69 6C 64  69 6E 67 73 2C 20 62 75 | ck buildings, bu
00FEB0: 69 6C 74 20 74 6F 20 70  6C 61 6E 2C 20 77 65 72 | ilt to plan, wer
00FEC0: 65 20 62 65 69 6E 67 20  62 75 69 6C 74 20 62 79 | e being built by
00FED0: 20 73 65 72 66 73 20 77  68 6F 73 65 20 6D 61 6E |  serfs whose man
00FEE0: 6F 72 69 61 6C 0A 6C 61  62 6F 72 20 77 61 73 20 | orial.labor was
00FEF0: 74 68 75 73 20 69 6E 63  72 65 61 73 65 64 2C 20 | thus increased,
```

```
00FF00: 74 68 6F 75 67 68 20 6C  65 73 73 65 6E 65 64 20 | though lessened
00FF10: 6F 6E 20 70 61 70 65 72  2E 20 48 65 20 64 69 64 | on paper. He did
00FF20: 20 6E 6F 74 20 6B 6E 6F  77 0A 74 68 61 74 20 77 |  not know.that w
00FF30: 68 65 72 65 20 74 68 65  20 73 74 65 77 61 72 64 | here the steward
00FF40: 20 68 61 64 20 73 68 6F  77 6E 20 68 69 6D 20 69 |  had shown him i
00FF50: 6E 20 74 68 65 20 61 63  63 6F 75 6E 74 73 20 74 | n the accounts t
00FF60: 68 61 74 20 74 68 65 20  73 65 72 66 73 E2 80 99 | hat the serfs’
00FF70: 0A 70 61 79 6D 65 6E 74  73 20 68 61 64 20 62 65 | .payments had be
00FF80: 65 6E 20 64 69 6D 69 6E  69 73 68 65 64 20 62 79 | en diminished by
00FF90: 20 61 20 74 68 69 72 64  2C 20 74 68 65 69 72 20 |  a third, their
00FFA0: 6F 62 6C 69 67 61 74 6F  72 79 20 6D 61 6E 6F 72 | obligatory manor
00FFB0: 69 61 6C 20 77 6F 72 6B  0A 68 61 64 20 62 65 65 | ial work.had bee
00FFC0: 6E 20 69 6E 63 72 65 61  73 65 64 20 62 79 20 61 | n increased by a
00FFD0: 20 68 61 6C 66 2E 20 41  6E 64 20 73 6F 20 50 69 |  half. And so Pi
00FFE0: 65 72 72 65 20 77 61 73  20 64 65 6C 69 67 68 74 | erre was delight
00FFF0: 65 64 20 77 69 74 68 20  68 69 73 00 7F 00 00 00 | ed with his....
```

```
010000: 20 76 69 73 69 74 0A 74  6F 20 68 69 73 20 65 73 |  visit.to his es
010010: 74 61 74 65 73 20 61 6E  64 20 71 75 69 74 65 20 | tates and quite
010020: 72 65 63 6F 76 65 72 65  64 20 74 68 65 20 70 68 | recovered the ph
010030: 69 6C 61 6E 74 68 72 6F  70 69 63 20 6D 6F 6F 64 | ilanthropic mood
010040: 20 69 6E 20 77 68 69 63  68 0A 68 65 20 68 61 64 |  in which.he had
010050: 20 6C 65 66 74 20 50 65  74 65 72 73 62 75 72 67 |  left Petersburg
010060: 2C 20 61 6E 64 20 77 72  6F 74 65 20 65 6E 74 68 | , and wrote enth
010070: 75 73 69 61 73 74 69 63  20 6C 65 74 74 65 72 73 | usiastic letters
010080: 20 74 6F 20 68 69 73 0A  E2 80 9C 62 72 6F 74 68 |  to his.“broth
010090: 65 72 2D 69 6E 73 74 72  75 63 74 6F 72 E2 80 9D | er-instructor”
0100A0: 20 61 73 20 68 65 20 63  61 6C 6C 65 64 20 74 68 |  as he called th
0100B0: 65 20 47 72 61 6E 64 20  4D 61 73 74 65 72 2E 0A | e Grand Master..
0100C0: 0A E2 80 9C 48 6F 77 20  65 61 73 79 20 69 74 20 | .“How easy it
0100D0: 69 73 2C 20 68 6F 77 20  6C 69 74 74 6C 65 20 65 | is, how little e
0100E0: 66 66 6F 72 74 20 69 74  20 6E 65 65 64 73 2C 20 | ffort it needs,
0100F0: 74 6F 20 64 6F 20 73 6F  20 6D 75 63 68 20 67 6F | to do so much go
```

```
010100: 6F 64 2C E2 80 9D 0A 74  68 6F 75 67 68 74 20 50 | od,”.thought P
010110: 69 65 72 72 65 2C 20 E2  80 9C 61 6E 64 20 68 6F | ierre, “and ho
010120: 77 20 6C 69 74 74 6C 65  20 61 74 74 65 6E 74 69 | w little attenti
010130: 6F 6E 20 77 65 20 70 61  79 20 74 6F 20 69 74 21 | on we pay to it!
010140: E2 80 9D 0A 0A 48 65 20  77 61 73 20 70 6C 65 61 | ”..He was plea
010150: 73 65 64 20 61 74 20 74  68 65 20 67 72 61 74 69 | sed at the grati
010160: 74 75 64 65 20 68 65 20  72 65 63 65 69 76 65 64 | tude he received
010170: 2C 20 62 75 74 20 66 65  6C 74 20 61 62 61 73 68 | , but felt abash
010180: 65 64 20 61 74 0A 72 65  63 65 69 76 69 6E 67 20 | ed at.receiving
010190: 69 74 2E 20 54 68 69 73  20 67 72 61 74 69 74 75 | it. This gratitu
0101A0: 64 65 20 72 65 6D 69 6E  64 65 64 20 68 69 6D 20 | de reminded him
0101B0: 6F 66 20 68 6F 77 20 6D  75 63 68 20 6D 6F 72 65 | of how much more
0101C0: 20 68 65 20 6D 69 67 68  74 20 64 6F 0A 66 6F 72 |  he might do.for
0101D0: 20 74 68 65 73 65 20 73  69 6D 70 6C 65 2C 20 6B |  these simple, k
0101E0: 69 6E 64 6C 79 20 70 65  6F 70 6C 65 2E 0A 0A 54 | indly people...T
0101F0: 68 65 20 63 68 69 65 66  20 73 74 00 80 00 00 00 | he chief st.•...
```

```
010200: 65 77 61 72 64 2C 20 61  20 76 65 72 79 20 73 74 | eward, a very st
010210: 75 70 69 64 20 62 75 74  20 63 75 6E 6E 69 6E 67 | upid but cunning
010220: 20 6D 61 6E 20 77 68 6F  20 73 61 77 20 70 65 72 |  man who saw per
010230: 66 65 63 74 6C 79 0A 74  68 72 6F 75 67 68 20 74 | fectly.through t
010240: 68 65 20 6E 61 C3 AF 76  65 20 61 6E 64 20 69 6E | he naïve and in
010250: 74 65 6C 6C 69 67 65 6E  74 20 63 6F 75 6E 74 20 | telligent count
010260: 61 6E 64 20 70 6C 61 79  65 64 20 77 69 74 68 20 | and played with
010270: 68 69 6D 20 61 73 20 77  69 74 68 0A 61 20 74 6F | him as with.a to
010280: 79 2C 20 73 65 65 69 6E  67 20 74 68 65 20 65 66 | y, seeing the ef
010290: 66 65 63 74 20 74 68 65  73 65 20 70 72 65 61 72 | fect these prear
0102A0: 72 61 6E 67 65 64 20 72  65 63 65 70 74 69 6F 6E | ranged reception
0102B0: 73 20 68 61 64 20 6F 6E  20 50 69 65 72 72 65 2C | s had on Pierre,
0102C0: 0A 70 72 65 73 73 65 64  20 68 69 6D 20 73 74 69 | .pressed him sti
0102D0: 6C 6C 20 68 61 72 64 65  72 20 77 69 74 68 20 70 | ll harder with p
0102E0: 72 6F 6F 66 73 20 6F 66  20 74 68 65 20 69 6D 70 | roofs of the imp
0102F0: 6F 73 73 69 62 69 6C 69  74 79 20 61 6E 64 20 61 | ossibility and a
```

```
010300: 62 6F 76 65 20 61 6C 6C  0A 74 68 65 20 75 73 65 | bove all.the use
010310: 6C 65 73 73 6E 65 73 73  20 6F 66 20 66 72 65 65 | lessness of free
010320: 69 6E 67 20 74 68 65 20  73 65 72 66 73 2C 20 77 | ing the serfs, w
010330: 68 6F 20 77 65 72 65 20  71 75 69 74 65 20 68 61 | ho were quite ha
010340: 70 70 79 20 61 73 20 69  74 20 77 61 73 2E 0A 0A | ppy as it was...
010350: 50 69 65 72 72 65 20 69  6E 20 68 69 73 20 73 65 | Pierre in his se
010360: 63 72 65 74 20 73 6F 75  6C 20 61 67 72 65 65 64 | cret soul agreed
010370: 20 77 69 74 68 20 74 68  65 20 73 74 65 77 61 72 |  with the stewar
010380: 64 20 74 68 61 74 20 69  74 20 77 6F 75 6C 64 20 | d that it would
010390: 62 65 0A 64 69 66 66 69  63 75 6C 74 20 74 6F 20 | be.difficult to
0103A0: 69 6D 61 67 69 6E 65 20  68 61 70 70 69 65 72 20 | imagine happier
0103B0: 70 65 6F 70 6C 65 2C 20  61 6E 64 20 74 68 61 74 | people, and that
0103C0: 20 47 6F 64 20 6F 6E 6C  79 20 6B 6E 65 77 20 77 |  God only knew w
0103D0: 68 61 74 20 77 6F 75 6C  64 0A 68 61 70 70 65 6E | hat would.happen
0103E0: 20 74 6F 20 74 68 65 6D  20 77 68 65 6E 20 74 68 |  to them when th
0103F0: 65 79 20 77 65 72 65 20  66 72 65 00 81 00 00 00 | ey were fre.◆...
```

```
010400: 65 2C 20 62 75 74 20 68  65 20 69 6E 73 69 73 74 | e, but he insist
010410: 65 64 2C 20 74 68 6F 75  67 68 20 72 65 6C 75 63 | ed, though reluc
010420: 74 61 6E 74 6C 79 2C 0A  6F 6E 20 77 68 61 74 20 | tantly,.on what
010430: 68 65 20 74 68 6F 75 67  68 74 20 72 69 67 68 74 | he thought right
010440: 2E 20 54 68 65 20 73 74  65 77 61 72 64 20 70 72 | . The steward pr
010450: 6F 6D 69 73 65 64 20 74  6F 20 64 6F 20 61 6C 6C | omised to do all
010460: 20 69 6E 20 68 69 73 20  70 6F 77 65 72 20 74 6F |  in his power to
010470: 0A 63 61 72 72 79 20 6F  75 74 20 74 68 65 20 63 | .carry out the c
010480: 6F 75 6E 74 E2 80 99 73  20 77 69 73 68 65 73 2C | ount’s wishes,
010490: 20 73 65 65 69 6E 67 20  63 6C 65 61 72 6C 79 20 |  seeing clearly
0104A0: 74 68 61 74 20 6E 6F 74  20 6F 6E 6C 79 20 77 6F | that not only wo
0104B0: 75 6C 64 20 74 68 65 0A  63 6F 75 6E 74 20 6E 65 | uld the.count ne
0104C0: 76 65 72 20 62 65 20 61  62 6C 65 20 74 6F 20 66 | ver be able to f
0104D0: 69 6E 64 20 6F 75 74 20  77 68 65 74 68 65 72 20 | ind out whether
0104E0: 61 6C 6C 20 6D 65 61 73  75 72 65 73 20 68 61 64 | all measures had
0104F0: 20 62 65 65 6E 20 74 61  6B 65 6E 20 66 6F 72 0A |  been taken for.
```

```
010500: 74 68 65 20 73 61 6C 65  20 6F 66 20 74 68 65 20 | the sale of the
010510: 6C 61 6E 64 20 61 6E 64  20 66 6F 72 65 73 74 73 | land and forests
010520: 20 61 6E 64 20 74 6F 20  72 65 6C 65 61 73 65 20 |  and to release
010530: 74 68 65 6D 20 66 72 6F  6D 20 74 68 65 20 4C 61 | them from the La
010540: 6E 64 20 42 61 6E 6B 2C  0A 62 75 74 20 77 6F 75 | nd Bank,.but wou
010550: 6C 64 20 70 72 6F 62 61  62 6C 79 20 6E 65 76 65 | ld probably neve
010560: 72 20 65 76 65 6E 20 69  6E 71 75 69 72 65 20 61 | r even inquire a
010570: 6E 64 20 77 6F 75 6C 64  20 6E 65 76 65 72 20 6B | nd would never k
010580: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 6E 65 77 | now that the.new
010590: 6C 79 20 65 72 65 63 74  65 64 20 62 75 69 6C 64 | ly erected build
0105A0: 69 6E 67 73 20 77 65 72  65 20 73 74 61 6E 64 69 | ings were standi
0105B0: 6E 67 20 65 6D 70 74 79  20 61 6E 64 20 74 68 61 | ng empty and tha
0105C0: 74 20 74 68 65 20 73 65  72 66 73 20 63 6F 6E 74 | t the serfs cont
0105D0: 69 6E 75 65 64 0A 74 6F  20 67 69 76 65 20 69 6E | inued.to give in
0105E0: 20 6D 6F 6E 65 79 20 61  6E 64 20 77 6F 72 6B 20 |  money and work
0105F0: 61 6C 6C 20 74 68 61 74  20 6F 74 00 82 00 00 00 | all that ot.◆...
```

```
010600: 68 65 72 20 70 65 6F 70  6C 65 E2 80 99 73 20 73 | her people’s s
010610: 65 72 66 73 20 67 61 76  65 E2 80 94 74 68 61 74 | erfs gave—that
010620: 20 69 73 0A 74 6F 20 73  61 79 2C 20 61 6C 6C 20 |  is.to say, all
010630: 74 68 61 74 20 63 6F 75  6C 64 20 62 65 20 67 6F | that could be go
010640: 74 20 6F 75 74 20 6F 66  20 74 68 65 6D 2E 70 72 | t out of them.pr
010650: 6F 6D 69 73 65 64 20 74  6F 20 64 6F 20 61 6C 6C | omised to do all
010660: 20 69 6E 20 68 69 73 20  70 6F 77 65 72 20 74 6F |  in his power to
010670: 0A 63 61 72 72 79 20 6F  75 74 20 74 68 65 20 63 | .carry out the c
010680: 6F 75 6E 74 E2 80 99 73  20 77 69 73 68 65 73 2C | ount’s wishes,
010690: 20 73 65 65 69 6E 67 20  63 6C 65 61 72 6C 79 20 |  seeing clearly
0106A0: 74 68 61 74 20 6E 6F 74  20 6F 6E 6C 79 20 77 6F | that not only wo
0106B0: 75 6C 64 20 74 68 65 0A  63 6F 75 6E 74 20 6E 65 | uld the.count ne
0106C0: 76 65 72 20 62 65 20 61  62 6C 65 20 74 6F 20 66 | ver be able to f
0106D0: 69 6E 64 20 6F 75 74 20  77 68 65 74 68 65 72 20 | ind out whether
0106E0: 61 6C 6C 20 6D 65 61 73  75 72 65 73 20 68 61 64 | all measures had
0106F0: 20 62 65 65 6E 20 74 61  6B 65 6E 20 66 6F 72 0A |  been taken for.
```

```
010700: 74 68 65 20 73 61 6C 65  20 6F 66 20 74 68 65 20 | the sale of the
010710: 6C 61 6E 64 20 61 6E 64  20 66 6F 72 65 73 74 73 | land and forests
010720: 20 61 6E 64 20 74 6F 20  72 65 6C 65 61 73 65 20 |  and to release
010730: 74 68 65 6D 20 66 72 6F  6D 20 74 68 65 20 4C 61 | them from the La
010740: 6E 64 20 42 61 6E 6B 2C  0A 62 75 74 20 77 6F 75 | nd Bank,.but wou
010750: 6C 64 20 70 72 6F 62 61  62 6C 79 20 6E 65 76 65 | ld probably neve
010760: 72 20 65 76 65 6E 20 69  6E 71 75 69 72 65 20 61 | r even inquire a
010770: 6E 64 20 77 6F 75 6C 64  20 6E 65 76 65 72 20 6B | nd would never k
010780: 6E 6F 77 20 74 68 61 74  20 74 68 65 0A 6E 65 77 | now that the.new
010790: 6C 79 20 65 72 65 63 74  65 64 20 62 75 69 6C 64 | ly erected build
0107A0: 69 6E 67 73 20 77 65 72  65 20 73 74 61 6E 64 69 | ings were standi
0107B0: 6E 67 20 65 6D 70 74 79  20 61 6E 64 20 74 68 61 | ng empty and tha
0107C0: 74 20 74 68 65 20 73 65  72 66 73 20 63 6F 6E 74 | t the serfs cont
0107D0: 69 6E 75 65 64 0A 74 6F  20 67 69 76 65 20 69 6E | inued.to give in
0107E0: 20 6D 6F 6E 65 79 20 61  6E 64 20 77 6F 72 6B 20 |  money and work
0107F0: 61 6C 6C 20 74 68 61 74  20 6F 74 00 82 00 00 00 | all that ot.•...
```


```
010600: 68 65 72 20 70 65 6F 70  6C 65 E2 80 99 73 20 73 | her people’s s
010610: 65 72 66 73 20 67 61 76  65 E2 80 94 74 68 61 74 | erfs gave—that
010620: 20 69 73 0A 74 6F 20 73  61 79 2C 20 61 6C 6C 20 |  is.to say, all
010630: 74 68 61 74 20 63 6F 75  6C 64 20 62 65 20 67 6F | that could be go
010640: 74 20 6F 75 74 20 6F 66  20 74 68 65 6D 2E 70 72 | t out of them.pr
010650: 6F 6D 69 73 65 64 20 74  6F 20 64 6F 20 61 6C 6C | omised to do all
010660: 20 69 6E 20 68 69 73 20  70 6F 77 65 72 20 74 6F |  in his power to
010670: 0A 63 61 72 72 79 20 6F  75 74 20 74 68 65 20 63 | .carry out the c
010680: 6F 75 6E 74 E2 80 99 73  20 77 69 73 68 65 73 2C | ount’s wishes,
010690: 20 73 65 65 69 6E 67 20  63 6C 65 61 72 6C 79 20 |  seeing clearly
0106A0: 74 68 61 74 20 6E 6F 74  20 6F 6E 6C 79 20 77 6F | that not only wo
0106B0: 75 6C 64 20 74 68 65 0A  63 6F 75 6E 74 20 6E 65 | uld the.count ne
0106C0: 76 65 72 20 62 65 20 61  62 6C 65 20 74 6F 20 66 | ver be able to f
0106D0: 69 6E 64 20 6F 75 74 20  77 68 65 74 68 65 72 20 | ind out whether
0106E0: 61 6C 6C 20 6D 65 61 73  75 72 65 73 20 68 61 64 | all measures had
0106F0: 20 62 65 65 6E 20 74 61  6B 65 6E 20 66 6F 72 0A |  been taken for.
```