**Archiving Challenge**

**BrieAnn Haberl**

**Buena Vista University**

**CMSC432 Operating Systems**

**Dr. Nathan Backman**

**Date: 12/08/2023**

**Identify Problem**

The main challenge involves creating two programs, as outlined by the final project doc: bvArchive and bvUnarchive. These programs should handle archiving and unarchiving directories and files. Much like zip files but without compression. I found this problem to be more interesting to me than the sorting challenge because it seemed easier with the amount of time for the final project.

* bvArchive: This program needs to open the directory of the directory name given, gather file metadata (names, sizes), and store that data in a structured file for archiving. It needs to store file contents as well.
* bvUnarchive: This program is responsible for doing the opposite of bvArchive. Reading an archive file, reconstruct the directory structure, and writing out the archived files with their original content and structure into the directory that was recreated.

**Identify Strategies**

The first thing I looked into was the libraries I would need to include for being able to interact with files and directories and get the necessary metadata and interactivity that would make this project easier. The next step was looking through the man pages of the libraries I was less familiar with to understand everything that is offered as built in options. Then I thought about some initial strategies for archiving information and unarchiving.

bvArchive:

* Set Tags: Set tags for the directory name and all of the files metadata (names, sizes, number of files “perhaps not needed if I have start and end tags of some kind” ).
* File Archiving: Structure the archived file format to preserve directory hierarchy and file content. Something along the lines of Directory Name, file name, file size #file start#, file contents, #file end# use this format to make it clear which file content belongs to which file so it can be recreated later.
* Number of Files: Keep track of the number of files to ensure they are all written back out This strategy guarantees that all files present are counted and at the very least you should be able to check that you end up with the same number of files when unarchiving
* Other Related Metadata: For the purpose of keeping this simple I do not think I will save things about file permissions and the time the folder was created since it will be creating it new with unarchive this being said there are many pieces of metadata that can prove useful that I may look into.

The structure of the test file

#DIRECTORY\_START#

sampleDir

#DIRECTORY\_END#

#FILE\_START#

C.txt

33

cccccccccccccccccccccccccccccccc

#FILE\_END#

#FILE\_START#

A.txt

17

aaaaaaaaaaaaaaaa

#FILE\_END#

#FILE\_START#

B.txt

9

bbbbbbbb

#FILE\_END#

The numbers being the bytes of the file

Having a clear way to read what the file consists of so that it can be recreated.

bvUnarchive:

* Archive Parsing: Parse the archived file, identify directory, and extract file metadata. Read in the directory name and create that directory first then to create the files it will know the start and end of each file and the file name and extension to recreate by using the tags that are set.
* Directory and File Creation: Recreate directory structures and restore archived files with their original content. To create them it just needs to be able to read everything in correctly to reformat the directory from the .data archive file. This relies on reading and parsing to be fully functional.

**Evaluate Solutions**

I feel like this plan while being pretty general is a good starting place for me to go about implementing a solution. I feel like sticking to a good format for my archive files will make it easier to keep things organized when it comes to reading and parsing information later. I feel like archive will be easier to implement than unarchive as it is just putting the files together instead of creating files. I did change strategies a bit for unarchive and ran into some issues with comparing strings for the names of directories and files so I restructured my archive a bit to make unarchiving easier.