CS311 - FA13: Archive

Trevor Bramwell

November 4, 2013

1 Design

This assignment is to create a simple implementation of the UNIX utility *ar*. The utility will support the operations on archive files of appending, creating, extracting, listing and deleting. It will also provide a flag for verbose output and if time permits, a way to add modified files, after a delay, from a directory.

Since this *ar* revolves around the previously mentioned operations, I will be structuring the code in the same manner. Depending on the flag passed in, the program will call the corresponding function to perform the operation. A set of utility functions will be created as well to reduce code duplication, and improve code in a more self-documenting way.

When I start writing my program, I will begin with the user interface, that is the command line. Having the interface created first will make it easier for me to focus on the functionality of the program.

I will then walk though each operation that can happen document the steps it needs to take, i.e. open archive file, write file, etc. to flesh out what the utility functions might be. I will then focus on the single operation of *listing*, and use a pre-created archive file so as to reduce the number of concerns. From there I will itertively enable each flag until they all work and the program works as intended.

1.1 Versions

Up to 0.5 GNU ar will be used for testing.

- 0.0 Tests (all flags, all flags w/verbose)
- 0.1 List (-t)
- 0.2 Extract (-x)
- 0.3 Append (-q)
- 0.4 Quick Append (-A)
- 0.5 Create Archive when using (-q, -A)
- 0.6 Delete (-d)
- 1.0 Verbose (-v) - Change all printf statements to only execute when flag passed.
- 1.1 Timeout (-w)

1.2 Deviation

After getting through appending and table, I realized verbosity was a simple if check away, and added it. Instead of moving onto extract and delete right way, I stuck with append and table until they were complete and free of bugs. I did not put aside enough time to complete extract and delete.

2 Questions

What do you think the main point of this assignment is?

The main point of this assignment is to get us reaquanted with the C programming language, and to learn UNIX system calls.

How did you ensure your solution was correct? Testing details, for instance. This section should be very thorough.

Originally I planned to check my program by using the python unittest library and subprocess modules. I could tell from the initial work on this that it would turn into a project on it's own and abandoned it in favor of manual testing.

The majority of my problems came from the even/odd padding in the ar program. It was a while before I realized I had been modding the size by 3 instead of 2 to check if the size was even or odd.

Most of my manual testing involved having a base archive pre-created with 'ar -cq' and then running my program off of that.

What did you learn?

I learned that I am not as strong of a C programmer as I though. A majority of challenges I encountered dealted with converting between types. I became familiar with sscanf and sprintf, but I still do not feel entirely comfortable with converting types in C.

I also learned how to use open, write, read, getopt, and became more familiar with C.

3 Work Log