ITP 435 Assignments

Command Line Arguments

Although we don't really use them much in this assignment, we want you to get comfortable with command line arguments. These are additional arguments you can pass in when running your program.

The way we've configured the project, you shouldn't edit `Main.cpp` to access the arguments. Instead, edit the `ProcessCommandArgs` function in `SrcMain.cpp`. (If you look at `Main.cpp`, you'll see it just directly calls `ProcessCommandArgs`).

The `argc` parameter is the number of command line arguments. This will always be `>= 1`. The `argv` parameter is an array of C-style strings (`const char *`). Each index contains a string for that argument.

The value of `argv[0]` is always the name of the executable (like "main.exe").

If `argc > 1`, then there are additional C-style strings like `argv[1]`, `argv[2]`, ..., `argv[argc - 1]`.

> **WARNING**
>
> In general, rather than directly manipulating the C-style strings (or doing comparisons with C-style strings), you should construct a `std::string` from the relevant argv and use that. For example, doing a comparison like `argv[1] == ".rl1"` does not compare the values of the strings but instead compares their pointers.

For this assignment, we only care if `argc == 2`. In this case, `argv[1]` contains the name of a file. If this file name ends with `".rl1"` then you should take the file and call `ExtractArchive` on it. For files that do not end in `".rl1"` call `CreateArchive`.

Assuming your Part 1 and Part 2 work, a correct `ProcessCommandArgs` implementations means you should get a 75/75 grade on GitHub Actions.

Once you're done, push your code to GitHub and confirm that your build succeeds and you pass all the graded tests. Refer to the grading rubric on the first page and make sure you satisfy the other criteria.