SIENA*college*
Computer Science

Computer Science 301
C Programming in Unix
Siena College
Fall A 2024

Assessment 1: Line Reverser

Due: 9:00 AM, Wednesday, September 18, 2024

In this assessment, you will be guided through the development of a C program that reverses the lines of an input file uses some of the language features we have studied so far.

You may work alone or in groups of size 2 or 3 on this assessment. However, in order to make sure you learn the material and are well-prepared for the exams, those who work in a group should either collaborate closely while completing the problems or work through the problems individually then discuss them within your group to agree on a solution. In particular, the "you do these and I'll do these" approach is sure to leave you unprepared for upcoming tasks and the exams and is prohibited.

Getting Set Up

In Canvas, you will find a link to follow to set up your GitHub repository, which will be named `revlines-assessment-yourgitname`, for this assessment. Only one member of the group should follow the link to set up the repository on GitHub, then others should request a link to be granted write access.

All GitHub repositories must be created with all group members having write access and all group member names specified in the `README.md` file by 11:59 PM, Friday, September 13, 2024. This applies to those who choose to work alone as well!

Programming Assignment

Write a C program that reads in lines from standard input, reverses each line, then prints it out. Note that you will be committing and pushing partial implementations, so pay careful attention to the directions. Since this is graded as a programming assignment rather than a practice program, be sure to read and follow the submission guidelines related to programming assignments.

You will need a `getline` function like the one in K&R on p. 29, but it needs to be named something else since it conflicts with the `getline` function from the C `stdio` library. Your repository has a copy of this function renamed as `get_line` implemented in `get_line.c` with a prototype in `get_line.h`.

The following steps guide you through the development – be sure to follow them in order, and complete each step before moving on.

1. Start by creating a new file `revlines.c`. Your file should include `get_line.h` to introduce the prototype for the `get_line` function. Write a `main` function that reads in lines from the input and prints them back out, continuing until the input ends. You may

assume a maximum line length of 1000 as in the K&R example. Your `main` function will be a stripped-down version of the `main` function on p. 29 of K&R (omit all the stuff about finding the longest line). Don't add any other functionality at this point!

2. Create a `Makefile`, along the lines of the one from the `gcd` example, that compiles and links your program into an executable called `revlines`. You can now compile, run, and test your program. It should echo back each line you enter. Typing `Ctrl-d` will signal end-of-input, which should terminate your program. **Commit and push this version with a commit message "testing input reading".**

3. Next, write and test a `reverse` function that reverses the contents of the string passed to it, in place. It should be implemented in `reverse.c` with a prototype in `reverse.h`. Do not use any C library functions in your implementation! This will be very similar functionally to the function or method you wrote previously that reverses the order of integers in an array up to but not including the first 0. Recall that C strings always end with a 0 character. Modify your `main` function to call `reverse` before it prints out each line. Also be sure to add the new file to your `Makefile`. **Commit and push this version with a commit message "added reverse".**

4. Your output probably looks a little strange, with a blank line printed between the text you type and the reversed version the program prints. Let's make two final changes: remove the `\n` in your `printf` in the idmain function, and don't include the new line character (`'\n'`) as part of the text that gets reversed by your `reverse` function. Your output should now look nicer. **Commit and push this version with a commit message "fix new line handling".**

5. If you have not been keeping up with the submission guidelines for programming assignments, take care of those details at this time, then commit and push your final version.

Run your program using the `whosonfirst.txt` file, redirecting the output to a new file `whorev.txt`. Add, commit, and push this output file to your repository.

Submission

Commit and push! (Which is already done if you've been following directions.)

Grading

This assignment will be graded out of 30 points.

Feature	Value	Score
Read lines until end of input	5	
`reverse` function correctness	10	
new line handling	4	
`Makefile`	1	
documentation, style, formatting	8	
`whorev.txt`	2	
Total	30	

There are no points awarded for including the 4 required Git commits, but a 15% penalty will be assessed for each missing commit.