CST 334: Operating Systems

Dr. Glenn Bruns

# MSH 4

**Purpose**. We are building our own shell to understand how bash works and to understand the Linux process and file API.

**Instructions.**  In this assignment we will add only one feature: redirection. To direct a command’s output to a file, the syntax “> outfile” is used. To read a command’s input from a file, the syntax “< infile” is used.

Your extended version of msh should extend the previous version of msh to handle commands like these:

$ ./msh

msh> ls -l > temp.txt

msh> sort < temp.txt > temp-sorted.txt

The result of these commands should be that the sorted output of “ls -l” is in file temp-sorted.txt.

Your shell builtins (like ‘cd’ and ‘help’) do not have to handle redirection. Only one new Linux command is needed: **dup2**.You will use dup2 for both input and output redirection. The basic idea is that if you see redirection on the command line, you open the file or files, and then use dup2. Think: do you want to do this in the child process that gets forked, or by the parent process?

Make sure you understand dup2. Please check out this [dup2 slide deck](https://drive.google.com/file/d/1-vlw93d4UUWGO5VabOFiI2IBi2nIJ0yd/view?usp=sharing) I created that explains dup2 and gives some hints on the homework.

Also, you need to handle the ">" and "<" characters that appear on the command line. Think about whether or not you put them into an array of tokens.

Testing your code: On mlc104, the directory /home/CLASSES/brunsglenn/cst334/hw/hw5/msh4 contains test files test1.sh and test2.sh and a Makefile. Copy these to the directory where you will develop your file msh.c. Each test should give exit status 0, like this:

$ ./test1.sh

$ echo $?

0

You need to run test1.sh first, as it will compile your code and produce binary file 'msh' that is used by the other tests. To use the Makefile, enter the command 'make' to run the tests. If you enter the command 'make clean', temporary files created by testing will be deleted.

**Submitting.** Submit your C file msh.c on iLearn.

**Grading**. 10 points will be given for each of 3 test cases, and 10 points will be given for neatly formatted and appropriately commented code. No “tidy code” points if your code isn’t close to being right. Your code should pass tests from previous weeks, plus any new tests related to redirection.