CS 241L - Data Organization

Spring 2022

Programming Assignment 1
Total points: 100
Due on September 1st, 2022

In this project, you will work in the Linux command line to write your first *C* programs. Follow the steps outlined below to complete your project and be sure to see the Grading Rubric provided at the end of this file as guide for completing the assignment.

Part 0: Extract files from tarball

1. The programming assignment is given to you as a compressed tarball. In order to extract the files from this tarball use the command `tar -xvf programming_assignment_1.tar.gz`. This will create a folder with name `'Programming Assignment 1'` in your current directory.

Part 1: Redirecting output from stdout to a file

1. Connect to a CS machine (moons.cs.unm.edu or b146 lab machines) using `ssh` with PuTTY, NoMachine, or the Terminal (Mac or Linux).
2. When you login, type `pwd`. You should be in your home directory which has the same name as your CS account username.
3. Create a directory called `project1` in your home directory.
4. Go to the `project1` directory using `cd`.
5. List contents of `project1` using `ls`. It should have no files and no subdirectories.
6. Open a new file called `shapes.c` with your editor.
7. `shapes.c` will contain the source code of a C program that prints the shapes in Figure 1:

   Hint: Modify the `hello_world.c` file to print these shapes provided to you.

Figure 1: These shapes are in the file `shapes.out` provided to you.

8. Use the command:

   `gcc -Wall -pedantic -ansi shapes.c`

   to compile and create an executable called `a.out`. It should compile without warnings or errors. If there are warnings or errors, edit your source file.

9. Look into the contents of `project1` with `ls`. Which files are there now?

10. Now use `ls` with an option to list both details (such as permissions, owner and group), and hidden files. What are these options? (Hint: use a dash and two letters after `ls`. Which letters are these?)

11. `ls` shows the contents of `project1` on the terminal screen. This is called the standard output stream, or `stdout` . You can redirect this output to a file if you would like to save it using the symbol `>` . Run the command:

    `ls > ls.out`

12. Now type `ls` and look at the list of files in `stdout` . You should see `ls.out` listed now.

13. Open `ls.out`. What are its contents? They should list the contents of running `ls`. Notice that `ls.out` is also included.

14. Now run the executable typing in the prompt:

    `./a.out`

15. You should see on the screen the output of the `shapes.c` program. If you do not see the shapes as displayed above, edit your source file.

16. Redirect output of C program. Combine the steps in (11) and (14) above to redirect the output of the shapes program to a file called `myshapes.out`. What do you observe on the terminal screen?

17. Type `ls` in the command prompt. Which files are there now? Update the `ls.out` file using redirection `(>)` again.

Part 2: ASCII and data types

1. The characters printed in Part 1 are ASCII characters with given values (see table from our lecture slides). The 7 characters printed in `shapes.out` are `- _ / \ : + *`

   What are the decimal values of the above characters?

2. Create a copy of your C program `shapes.c` with `cp` and name it `newshapes.c`.

3. Modify `newshapes.c` so that it prints the same image as in `shapes.out` but uses different characters. The ASCII decimal values of the new characters are the decimal values of the original characters (listed above in 1) **minus** 9. For instance, - has a 45 value in ASCII, so the new character would be the one whose value is 36. This new character will be printed where all - are placed originally in `shapes.out`.

   Hints:

   - Make -9 a constant in your program called SHIFT using `#define` or `const`
   - Keep your program clear and concise by declaring new variables when needed.

4. Compile `newshapes.c` with the `-Wall -pedantic -ansi` options.

5. Run your new `a.out` executable and redirect its output to a file called `mynewshapes.out`. Note that an output file is **not** provided to you for comparison.

What to submit:

Submit a tarball to Canvas with name `<Your-Student_ID>_1.tar.gz` containing the following files:

`ls.out`

`shapes.c`

`myshapes.out`

`newshapes.c`

`mynewshapes.out`

The command for this is `tar -czvf <Your-Student_ID>_1.tar.gz ls.out shapes.c myshapes.out newshapes.c mynewshapes.out`.

Grading Rubric:

If any of your C programs do not compile with the `-ansi -pedantic -Wall` options without errors or warnings the points given for the assignment will be zero. Otherwise the following rubric will be used:

+ 7 pt: Your C programs start with a comment on top of the file with your name and description of the programs

+ 10 pts: Your C programs follow the class coding standards

+ 10 pts: You have submitted the required files separately (you did ***not*** combine them into a single zip file).

+ 27 pts: Your output file `myshapes.out` passes a `diff` test when compared with the output file `shapes.out` that was provided to you.

+ 27 pts: Your output file `mynewshapes.out` passes a `diff` test when compared with an output file that only the grader has access to.

+ 19 pts: Your program `newshapes.c` produces the correct output when the grader changes the value SHIFT to a different number where it is declared or defined.