Assignment 0: Assignment Workflow
CSE 130: Principles of Computer Systems Design

Due: October 8, 2024 at 11:30 PM

Goal This project will familiarize you with the development environment and submission practices in CSE 130. There is very little code for you to produce, but I encourage you to start early nonetheless.

Assignment Details

For this assignment, you will write a program, `hello`, that produces the text `Hello World!` to the terminal via `stdout`. Additionally, you will install and use an Ubuntu 24.04 virtual machine on your local machine so that you can test your code locally. You will attest to having installed a virtual machine by including the text `"I successfully created and am using an Ubuntu 24.04 virtual machine."` in a file named `attestation.txt` [1]

Rubric

We will use the following rubric for this assignment:

Category	Point Value
Makefile	10
Clang-Format	5
Files	5
Functionality	10
Total	30

Makefile Your repository includes a Makefile with the following rules:

- `all`: produces the `hello` binary.
- `clean`: removes all `.o` and binary files.
- `hello`: produces the `hello` binary.

Additionally, your Makefile should use clang (i.e., it should set `CC=clang`), and should use the `-Wall`, `-Wextra`, `-Werror`, and `-pedantic` flags (i.e., it should set `CFLAGS=-Wall -Wextra -Werror -pedantic`).

Clang-Format All `.c` and `.h` files in your repository are formatted by the `.clang-format` file included in your repository.

Files The following files are included in your repository:

- `hello.c`
- `Makefile`
- `README.md`
- `attestation.txt`

Additionally, no binary files nor any object files (i.e., `.o` files) are included in your repository.

[1] We realize that you could lie about your virtual machine. But, we will assume that you have one installed and working in future assignments.

Functionality There are two "functionality" components to this assignment, each worth equal weight:

1. `hello`. `hello` should write `"Hello World!"` to `stdout` when invoked. It must be written using the 'C' programming language (*not C++!*). It cannot use the following functions from the 'C' `stdio.h` library: `fwrite`, `fread`, variants of `put` (i.e., `fptuc`, `putc`, `putc_unlocked`, `putchar`, `putchar_unlocked`, and `putw`), and `get` (i.e, `fgetc`, `getc`,`getc_unlocked`, `getchar`, `getchar_unlocked`, and `getw`). You cannot use functions, like `system(3)`, that allow you to execute external programs.
2. `attestation.txt`. Your repository includes the file, `attestation.txt`, which states the *exact* text: `"I successfully created and am using an Ubuntu 24.04 virtual machine."`.

How to submit

1. Navigate to the `asgn0` directory in your repo.
2. Configure your `config.json` file (see Canvas or the README in the resources repo for more information).
3. `python3 -m autograder.run.submit [file1] [file2] ...`

Resources

Here are some resources to help you:

Testing We will provide you with two resources to track your progress and see how well you are doing on the above rubric. First, an autograder, which you will run locally through the command line.

Second, we will supply you with a set of test scripts in the resources repository to check your functionality. You can use the tests to see if your functionality is correct by running them on your Ubuntu 24.04 virtual machine. For this assignment, we will provide you with all of the tests. In future assignments, we will provide you with a subset of the tests that we will run.

Hints Here are some hints to help you get going:

- Check out the document "Installing Multipass" on Canvas. It has some instructions for getting an Ubuntu 24.04 virtual machine up and running.
- To format a file, named `foo.c`, using the included `.clang-format`, execute the command `clang-format -i -style=file foo.c`. If you include the `.clang-format` file in the root directory of your repository (as it was when we created your repositories), you can instead execute `clang-format -i foo.c`.