Notes for ECE 26400 - Advanced C Programming

Ezekiel Ulrich

August 24, 2023

These are lecture notes for fall 2023 ECE 26400 at Purdue. Modify, use, and distribute as you please.

Contents

Course Introduction

This class will be taught by Prof. Joy Xiaoqian Wang. There will be four online exams, weekly online quizzes, and 20 homework assignments. For more information, consult the syllabus here.

Tools

UNIX System: The environment we'll use in this course. No matter your machine, you can use the UNIX environment. Some common commands in UNIX-like systems are:

For a comprehensive list of UNIX commands, see Wikipedia's excellent page

- `ls` - List directory contents
- `cd` - Change directory
- `mkdir` - Make directory
- `rm` - Remove files or directories. Use -rf to recursively delete files regardless of permission
- `mv` - Move files or directories
- `diff` - Comparing two files and showing their difference. Use -w to ignore whitespace
- `cat` - Shows contents of file without opening
- `cp` - Create a copy of a file
- `[CTRL + U]` - Undoes what was last typed
- `chmod` - Change file permissions
- `chown` - Change file ownership

- `kill` - Terminate processes
- `ssh` - Secure shell remote login
- `scp` - Securely copy files between hosts
- `wget` - Download files from the web
- `find` - Search for files and directories
- `vim` - Powerful text editor

To use these, simply type them in bash. For example, ls will print the contents of your directory.

Listing 1: Using ls

```
$ ls
code-folder/    helloworld.c    homework/
```

Git for Version Control: Distributed version control system. Git helps you manage, store, and collaborate on your project. The "version control" refers to how Git stores previous versions of your code, so unwanted changes can be reverted. Git is useful for when several people work on a project at a time or when you want to keep track of changes.

When using Git, you will have a staging area on your computer where you directly edit your code, a local repository (or repo) that tracks all the files associated with a project, and a remote repo to store the project. For this class, the remote repo is what's that's graded (sending TAs to check student computers took too long).

You code on your local repo and then push it to the remote repo. We can also pull updates form the remote repo to your local.

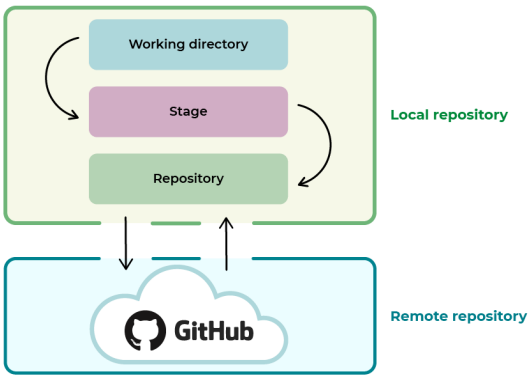

Figure 1: Layout of staging area, local repo, and remote repo

Some common Git commands are:

- `git push` - Replace what's on the remote repo with your local repo
- `git pull` - Replace your local repo with the remote repo
- `git init` - Creates a new Git repository
- `git clone` - Gets repo from specified url and copies to your machine, creating a new local repo
- `git add` - Adds file to staging area
- `git status` - Check what files in the working directory are added or committed
- `git log` - Check different versions of each project
- `git commit -m` - Moves changes from staging area to local repo. Use -m to add a message, and push to remote repo with `git push`
- `git reset` - Resets local repo to earlier version

GCC: Compiles C code to executable program. Compiling a file with gcc is simple:

Listing 2: Using gcc

```
$ gcc homework-one.c
```

Here are some useful gcc options:

- `gcc [filename] -o [output name]` - Change executable file name
- `gcc -g [filename]` - Generates debug information to be used by GDB debugger.
- `gcc -Wall` - Enables all compiler's warning messages. This option should always be used, in order to generate better code.

Makefile: Allows us to specify which options should be used when gcc is called.

Listing 3: Makefile

```
GCC=gccc
CFLAGS=-std=c99 -g -Wall -Wshadow --pedantic -Wvla -Werror
EXEC = sort
TESTFLAGS = -DASCENDING

all: main.c sort.c
	$(GCC) $(CFLAGS) -o $(EXEC) main.c sort.c

# Fix the gcc command for the target ascending by fill in the missing part
ascending: main.c sort.c
	$(GCC)
```

Called with male all in bash

GDB: