Lecture 2 - 6/1/2022

Introduction

Nelson has posted videos from previous semesters in Panopto incase you want to watch them before class.

Ekesh Kumar's notes is available on the class website

Review

When we log into Grace, everyone's environment looks different because everyone has different directories and things in their environment.

At this point, you should have run the setup command.

- This will give you the `216` and `215public` directories.
    - the `216public` directory is read-only, so students cannot change it.

We will work on projects and things in the `216` directory. The `216public` directory is read-only and it is where class materials are posted.

The colors in Nelon's screen are due to the usage of MobaXTerm which has text highlighting.

This class has very basic unix commands

- cd
- ls
    - ls -F shows slash next to all the folders
- mkdir

When you connect to Grace, you will see a number next to `Grace` such as `grace3:`. This number represents the specific host within the cluster to which you have connected.

Copying Things

This command copies the `Week01` directory from the `216public` directory to our own `216` directory in our home directory.

```
cp -r ~/216public/lecture_examples/Week01/ ~/216
```

- `cp` is the copy command.
- `~` (tilde) represents the home directory.
- `-r` means recursive
- `~/216public/lecture_examples/Week01/` is what we want to grab.
- `~/216` is where we want to put it.

Note: pressing tab after typing the beginning of a word or command will autocomplete it. It will even autocomplete using the names of the contents which you can access.

Note: Since `~` represents the home directory, typing `cd ~` is the same as typing `cd`.

Removing things

This command removes the `Week01` and everything in it.

- Unix does not give any second chances. You cannot easily recover a deleted file.

```
rm -r Week01
```

To bypass all the confirmations, use `-f`, which means force.

Moving things

Suppose we want to move things from a directory `old` to `old2`.

The command `mv` moves things.

```
mv old old2
```

Typing in files

To type in a file, we can simply use an editor such as vi, vim, or nano. For example, we can type in a file called `p1.c` by using

```
vi p1.c
```

Then we can simply type code such as this:

```
#include <stdio.h>

int main() {
    int x = 100;

    printf("%d\", x + 2);

    return 0;
}
```

To save your work, hit escape, then hit `shift` + `:`, then type `w` and then `enter`.

- `shift` + `:` escapes the typing section of `vi` and lets you enter commands.
- the command `w` denotes write changes to file
- the command `q` denotes quit `vi`.

To summarize:

- Escape using `shift+:`.
- Write the file using `w`.
- Quit the editor using `q`.
- Commands can be chained together such that `wq` will write your changes to the file and subsequently quit `vi`. Often, we will write and quit so we just do `wq` in vim.

These commands are the same in `vim`.

More commands:

- `o` adds a line

Class Website

On the class website, you can see resources by going to the dropdown menu. Here, you will find all the Linux commands you will need for this class.

Aliases

When one wants to compile code in this class, we must use `gcc` with many flags like this:

```
gcc -ansi -Wall -g -O0 -Wwrite-strings -Wshadow -pedantic-errors -fstack-
protector-all -Wextra p1.c
```

This is a rediculous amount of flags to write, so we can create an alias for `gcc` such that whenever we use `gcc`, all those flags are used. To do this, follow the instructions here.

To know if you did the alias correctly, you will know if you pass the class.

- just kidding. You can check by typing `alias gcc`

C

Let's take a look at `print example` in `C-Language-II-Code` in `216public`.

```c
#include <stdio.h>

int main(void) {   /* Notice the void to indicate no parameters */
   int age = 18;
   float salary = 100.50;
   char gender = 'F';
   const char *address = "AV Williams Bld";

   printf("Age: %d, Salary: %f, Gender: %c\n", age, salary, gender);
   printf("Address: %s\n", address);

   return 0;
}
```

The line

```
printf("Age: %d, Salary: %f, Gender: %c\n", age, salary, gender);
```

is a formatted string.

Next, let's look at the program `reading.c`.

In this code, there is `%c` and `%d` for input.

- `%d` skips/ignores all spaces and newlines.
- `%c` inludes spaces and newlines

scanf() conversion specifiers

- `%d`, `%x`, `%o`
  - Reads a decimal, hex, or octal number into an `int`
- %f reads in a `float`
- %lf reads in a `double`
- %ld reads in a `long`
- %c reads in a `char`
- %s reads in a `string` (bounded by whitespace)

Naming Convention in C

In C, we don't use camelCase by convention. Instead, we separate words with underscores. Instead of `letterGrade`, we would use `letter_grade`.