CSE 351 Section 1: Intro, C Programming, C Tools

Katelin Bailey
University of Washington

Introduction

Katelin Bailey

- 3rd Year PhD Student in Computer Science and Engineering
- Research in the co-evolution of Hardware and Software Systems (e.g. the OS)
- Office Hours: Today! at 1:30 in 218
- Contact: katelin@cs or find me in 591

Course Tools

Use what works for you

- ssh [username]@attu.cs.washington.edu
- download a home VM
- use lab machines
- use a home install

Course Tools

If you use the home VM...

- Connect to the UW or CSE network to download: it's 6 GB and it will take a while

- The instructions online are easy to follow when you find them.

- If you run into trouble, email me or the course staff for help (We've all done it)

Course Tools

Unix! Get comfortable with it.

- cd: change directory (goto)
  - `cd /path/to/directory`
  - `cd ~`
- pwd: print working directory (where am I?)
- ls: list directory contents (what's in here)
- chmod: change mode (permissions)

We'll assume that you know basic unix commands. If you need help, there are some online guides...

Course Tools

If you need help with any of these topics, look for cheatsheets to be posted later today:

- Quick and Dirty Guide to C
- Unix for Newbies
- Emacs Shortcut Cheatsheet

Also try online guides and the man pages!

Course Tools

Find an editor that works for you, and master it

- emacs, vi, gedit, eclipse
- the important thing is that you know how to use your editor of choice
- basic familiarity in emacs and vi are great (even if you like a gui to edit)

C Code

If this is your first introduction to C...

- Don't be overwhelmed
- Make sure you know, use, and play with pointers
- Ask for help! There are parts of C that are not intuitive.

Compiling C Code

If you haven't looked at the Lab 0 yet...

- We compile with GCC (gnu c compiler)
- Available on most machines

There are two steps:

- Compile: `gcc -c example.c`
- Link: `gcc -o example example.o`

Compiling C Code

There are lots of ways to combine compiling and linking, and lots of options to add. For this course, you'll be adding:

- `-g` for debugging info
- `-Wall` to warn about known issues
- `-std=gnu99` to use the latest (1999) standard
- All together: `gcc -o example example.c -g -Wall -std=gnu99`

Hello World

A Basic C Program

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char*argv[]){
  printf("Hello, World!\n");
  return 0;
}
```

Hello World

A Basic C Program

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char*argv[]){
  printf("Hello, World!\n");
  return 0;
}
```

Hello World

Headers are special C files

- They provide declarations of other code
- stdio and stdlib are the two basic ones, pretty much always include them
- basic header files are include in /usr/include
- you can add your own if you provide a path to it (`#include "path/to/include.h"`)

Hello World

A Basic C Program

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char*argv[]){
  printf("Hello, World!\n");
  return 0;
}
```

Hello World

The main function has three aspects:

- First parameter (argc) is the number of arguments from the command line
- Second parameter (argv) is an array of strings: arguments
- That means that argc is the number of elements in argv: very useful
- Lastly, a return code indicates success or failure.
    - Why return 0?

Hello World

A Basic C Program

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char*argv[]){
  printf("Hello, World!\n");
  return 0;
}
```

Hello World

Printing in C:

- `printf("I am an output string\n");`
- equivalent to System.out.printf()
- special format codes allow you to print variables
- `printf("int: %d\n", integer_value);`
- Lookup the format codes for floats, hex, strings, etc. You'll need them to debug and display output.
- Why these ugly formats? Because string concat is hard in C

Man Pages

- Use them. Here's an example:

Debugging in C

You can use printf, and sometimes you need that... but it's really inefficient

- Use GDB instead!
- GDB: Gnu Debugger (it's magic)
- Remember to compile with `-g`
- Then run your program inside of gdb
  - Here's an example...

Debugging in C

There a lot of ways to use GDB...

- We'll be posting a video online
- There are cheatsheets all over the internet
- Explore: you'll find what works in your programs.

Lab 0

If we have time... here's how to go through lab0