CS 241: Systems Programming

Lecture 1. Introduction

Fall 2025

Prof. Stephen Checkoway

What is this course about?

Tools for succeeding in computer science

- Unix command line
- Bash scripting
- **Rust programming**
- Building software
- Debugging
- Linters/static analyzers
- Version control
- Collaborative development
- Regular expressions
- Looking things up like a real programmer
- …

What is this course not about?

Learning to program (you already know how to do that!)

…but you'll get a lot of good programming practice and level up as a programmer

You should expect to

Do a lot of programming and debugging!

Learn about tools by reading their documentation and searching the Internet

Read a lot

Work on projects in groups

What is “Systems Programming”?

Systems programming

- Builds software to support other programs (e.g., operating systems, network services)
- Traditionally written in low-level languages (first assembly, then C)
- Often written with specific computer hardware in mind (e.g., x86-64 or ARM64)

Contrast with application programming which

- Builds software for end users (e.g., word processors, video players)
- Often built in languages with automatic memory management like Java, C#, Swift

We’ll be learning Rust, a general purpose programming language good for both!

Who am I? Professor Stephen Checkoway

- Research: Computer security, unexpected computation

- Fun Facts:
  - I’m face blind
  - I have three Oberlin cats

Course Policies

- We'll go over some policies today (grading, clickers, assignments, etc.)
- **READ THE SYLLABUS** for full details on course policies (e.g., schedule, accessibility, etc.)
- I will primarily use Ed announcements to communicate any changes or updates with you (as well as mentioning in class)

Grading

- 65% Weekly labs
- 15% Final group project
- 10% Class participation
- 10% Reading exercises

The final project must be completed to pass the course

Clickers!

- Lets you vote on multiple choice questions in real time.
- Like pub trivia, except the subject is always systems programming.
- You need one by next Wednesday

iClicker Accounts

- You must create an iClicker account to ensure your grades are counted
- Visit iClicker.com > Create an Account > Student.
- Or, download the iClicker Student iOS/Android app. Select Sign Up! to create your account.
- If you already have an iClicker account, just sign in! Do not create and use more than one iClicker account as you will only receive credit from a single account.
- You should have received an email from me/iclicker with details on how to do this
- If you have a physical iclicker, you do not need to pay anything for the account – just enter the iclicker id

Peer instruction

I will pose a multiple-choice question about a concept

Think and choose your answer individually with your clicker

After the time ends, discuss your answers with the people around you, come to consensus, and vote again

After the group vote, you explain why your group voted that way

Why peer instruction?

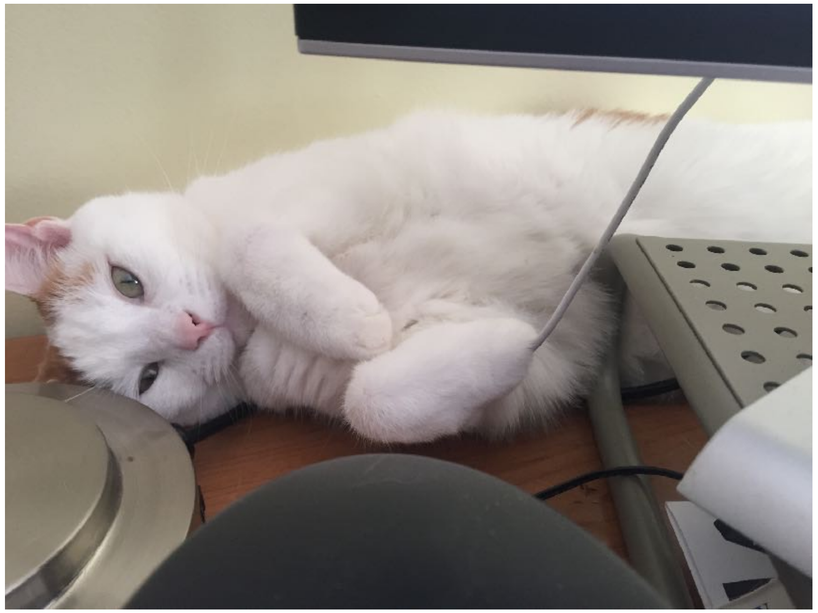
You get to make sure you're following the material

I get immediate feedback about what parts are confusing

It's less boring than lecture

Research shows it promotes more learning than standard lecture

Which cat is cutest?



A. Kirk

B. Bones

C. Equally cute

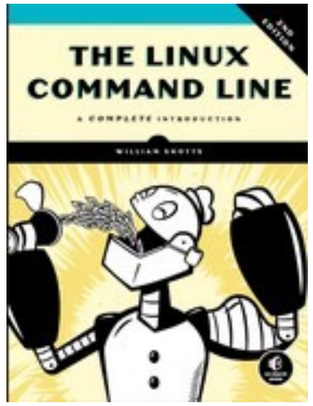
D. I don't like cats (I'm a monster)

Reading

- Due before each class
- You need to do the reading and then answer three multiple choice questions about it on GradeScope (linked from BlackBoard)
- First reading due Friday
- All textbooks are available for free online

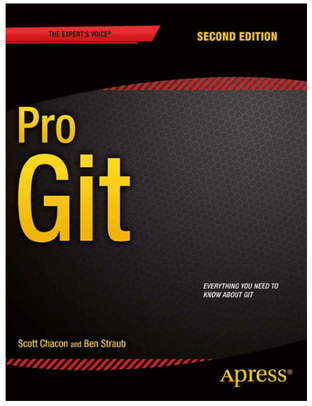
Textbooks

William E. Shotts. *The Linux Command Line*, 2nd edition



Steve Klabnik and Carol Nichols. *The Rust Programming Language*

Scott Chacon and Ben Straub. Pro Git, 2nd edition



Labs

First one (Lab 0) is Tuesday (it will be short)

Subsequent labs will be longer (like 150/151 labs) and have code to turn in each week

Final few labs of the semester will be designed to be completed in the lab period (so you can work on your group project)

Lab room

King 201

You should have 24 hour swipe access to both this lab and King 135/137

Late days/missing class

You have 3 late days you can use on any homework

- If you work with a partner (and you should), late work counts against both of your remaining late days

Late assignments will not be accepted after the 3 late days are used unless you have discussed with me

Final group project

Work in groups of 3-4

More about this in a few weeks

You will write a project proposal

You will have a bunch of time to work on it

You will give a short presentation on it at the end of the semester

Everybody is expected to work on all parts of the final; in particular, if you do not show up for the presentation, you will fail the course

Honor code

Do

- Work in groups of size 2 (or 4 for the project)
- Discuss assignments with others in the class, including on the discussion forum
- Cite sources if using code/ideas from outside class

Do not

- Share your solutions outside your group
- Use someone else's solutions
- Use ChatGPT/Github Copilot
- Give your clicker to someone else if you’re not present in class

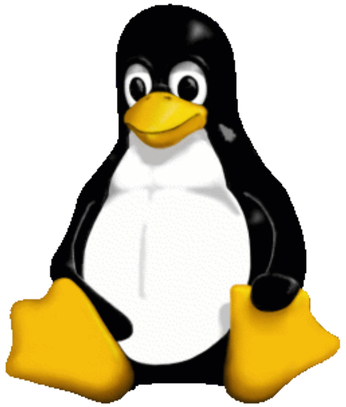
Syllabus

On course website (linked from BlackBoard)

Connecting to mcnulty

Remote server for CS 241: mcnulty.cs.oberlin.edu (named for Kay McNulty)

- Access via ssh
- Accessible from on campus but not outside Oberlin



Windows

From a terminal:

Use PuTTY https://putty.org/

```
$ ssh username@mcnulty.cs.oberlin.edu
```

Can't connect to mcnulty?

mcnulty.cs.oberlin.edu isn't reachable outside of Oberlin

occs.cs.oberlin.edu is!

```
$ ssh username@occs.cs.oberlin.edu
$ ssh mcnulty.cs.oberlin.edu
```

Alternatively

```
$ ssh -J user@occs.cs.oberlin.edu user@mcnulty.cs.oberlin.edu
```

Editors

For the Rust portion of the course, we'll be using Visual Studio Code

For the Bash portion of the course, you can use whatever editor you like

- Learning a command-line editor like vim or emacs can be really helpful
- Using VS Code is also an excellent choice

Video Studio Code lets you connect to computers via ssh for remote editing

Why Rust

Traditionally 241 taught C

C is (essentially) impossible to write correctly and incorrect C leads to vulnerabilities which can be exploited

C does almost nothing to prevent you from writing unsafe programs

Rust is the future of systems programming, here's just a taste

- Ubuntu just adopted a Rust implementation of sudo
- Rust code is appearing in Linux and Windows operating systems
- ARM is writing some low-level firmware in Rust
- Python's cryptography module is written in Rust

A Python program

```
small_primes = [2, 3, 5, 7, 11, 13, 17, 19, 23]
index = int(input("Enter the index of a small prime: "))
print(f"The small prime at index {index} is {small_primes[index]}")
```

```
$ python small_primes.py
Enter the index of a small prime: 4
The small prime at index 4 is 11
```

What happens if I enter a number that's too large?

```
$ python small_primes.py
Enter the index of a small prime: 10
```

A Python program

```
small_primes = [2, 3, 5, 7, 11, 13, 17, 19, 23]
index = int(input("Enter the index of a small prime: "))
print(f"The small prime at index {index} is {small_primes[index]}")
```

What happens if I enter a number that's too large?

```
$ python small_primes.py
Enter the index of a small prime: 10
Traceback (most recent call last):
  File "/Users/steve/small_primes.py", line 3, in <module>
    print(f"The small prime at index {index} is
{small_primes[index]}")
IndexError: list index out of range
```

A Java program

```
public class SmallPrimes {
    public static void main(String[] args) {
        int[] small_primes = {2, 3, 5, 7, 11, 13, 17, 19, 23};
        System.out.print("Enter the index of a small prime: ");
        System.out.flush();
        int index = Integer.parseInt(System.console().readLine());
        System.out.println("The small prime at index " + index +
                           " is " + small_primes[index]);
    }
}
```

```
$ java SmallPrimes
Enter the index of a small prime: 4
The small prime at index 4 is 11
```

What happens if I enter a number that’s too large?

A Java program

```
public class SmallPrimes {
    public static void main(String[] args) {
        int[] small_primes = {2, 3, 5, 7, 11, 13, 17, 19, 23};
        System.out.print("Enter the index of a small prime: ");
        System.out.flush();
        int index = Integer.parseInt(System.console().readLine());
        System.out.println("The small prime at index " + index +
                           " is " + small_primes[index]);
    }
}
```

What happens if I enter a number that's too large?

```
$ java SmallPrimes
Enter the index of a small prime: 10
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
  at SmallPrimes.main(SmallPrimes.java:7)
```

A C program

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

int main(void) {
    int small_primes[] = {2, 3, 5, 7, 11, 13, 17, 19, 23};
    printf("Enter the index of a small prime: ");
    fflush(stdout);
    char line[1024];
    fgets(line, sizeof line, stdin);
    int index = atoi(line);
    printf("The small prime at index %d is %d\n", index,
            small_primes[index]);
    return 0;
}
```

A C program

```
$ ./small_primes
Enter the index of a small prime: 4
The small prime at index 4 is 11
```

What happens if I enter a number that’s too large?

```
$ ./small_primes
Enter the index of a small prime: 10
```

A C program

```
$ ./small_primes
Enter the index of a small prime: 4
The small prime at index 4 is 11
```

What happens if I enter a number that's too large?

```
$ ./small_primes
Enter the index of a small prime: 10
The small prime at index 10 is 129070014
```

???

```
$ ./small_primes
Enter the index of a small prime: -7
The small prime at index -7 is 83427584
```

???

For next class

Register your iClicker

Read chapters 1–5 of *The Linux Command Line*; **complete reading quiz by the start of class on Friday**

Reminder: **We have lab next Tuesday! (1:00 p.m., King 201)**