

Welcome

211!

# Welcome to Systems Fundamentals

- This course is designed around three content areas:

1. Systems Programming

2. Representation Translations

3. Tools for Computer Scientists

# Where does this course fit in at UNC?

- **COMP211** - **Systems Fundamentals**

# Who am I?

- Kris Jordan
- Teaching Professor since 2015
  - UNC with a BS in Computer Science c/o 2007
  - Brown University Master's c/o 2008
- This course was a missing course for me when I graduated...
  - ... and it's been a missing course in the department since.
- I'm *really* excited to bring this course to life at UNC!

# Meet your Team

Alana Fiordalisi

Chris Diserafino

Cindy Wang

Dylan Binley

Eric Schneider

Hanna Tischer

Jules Perkins

Larry Li

Maddie Huber

Matthew Kirby

Milen Patel

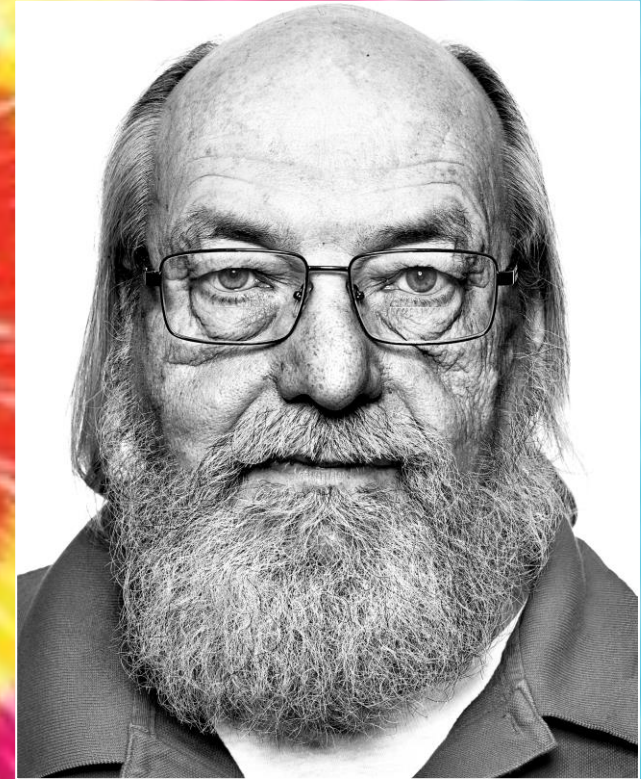
San Bae



# Protagonists in our Hero's Journey through the Psychedelic Little Languages of \*Nix



Lorinda Cherry  
aka Pipeline Queen



Ken Thompson  
aka Gandalf of  
Unix

# How to ask questions:

- Login to PollEverywhere: **[polleverywhere.com/auth/saml/unc](https://polleverywhere.com/auth/saml/unc)**
- Questions can be asked at: **[PolleEv.com/compunc](https://PolleEv.com/compunc)**

# 1. Exploring Systems Programming

- What are the foundations beneath programming language concepts?
  - data representation
  - machine instructions
  - pointers
  - runtime environments
  - execution models
- What are the concerns of *a process* in a **system of many processes**?
  - process model
  - memory management
  - systems / API calls
  - input / output



## 2. Exploring Representation Transformation

- How are *data* and *code* translated to representations for processing?
  - regular expressions
  - lexical analysis
  - parsing
  - code generation vs. evaluation
- What are some of the *fundamental translators* of the field?
  - compiler vs. interpreter vs. bytecode virtual machines
  - assembler
  - linker / loader

# 3. Exploring Tools for Computer Scientists

- What are the tools used by professionals in our field?
  - command-line interface shells
  - text editors
  - compilers and linkers
  - test harnesses
  - debuggers
  - version control software
  - workflow automation
- What are *best practices* of today's software engineering teams?
  - Program style and code review
  - Test-driven development
  - Pair programming
  - Collaboration in version-controlled projects

# Why is this course important to your **career**?

- Computer scientists and software engineers depend on **tools** in their day-to-day work.
- Effectively **wielding tools** and occasionally **inventing your own** are force multipliers when solving real world problems.
- My goals for this course are for you to:
  - Embrace how **well-designed abstraction layers, APIs, and composition** enable whole systems to be greater than the sum of their parts.
  - Gain practice **learning new languages** and tools on your own.
  - Be prepared to **start, organize, and tool your own software engineering projects**.

# Little Languages are a central theme...

"Little Language" is a term coined by Jon Bentley (UNC '76):

"Languages surround programmers, *yet many programmers don't exploit linguistic insights.*

Examining programs under a linguistic light can give you a better understanding of the tools you now use,

and can teach you design principles for building elegant interfaces to your future programs."

# Little Languages we will encounter...

Bash Shell Scripting Language

The C Programming Language

Regular Expressions

Makefiles

Markdown

*... potentially more!*

# This course will not teach you *a//* the tools...

- Or even *most* of them. Knowing them all, like knowing every programming language or every spoken language, is impossible.
- This course will teach you how to think about little languages, their structure, and how to implement your own.
- We will encounter many enduring, brilliant, and historically significant little languages most computer scientists and software engineers make use of **frequently** today...  
...most of which were invented in the **70s**!
  - Thus, you *will* leave with a solid handle on important tools that can improve your productivity and enable you to automate tedious tasks away.



# How does this course relate to others?

- Every Course
  - Every course has tools at work behind the scenes. Some hide them more than others. The more you understand about how they work they more effective you'll be at solving problems.
- COMP431 – Networking
  - Protocols are little languages that need to be parsed
- COMP455 – Theory of Automata
  - We will make concrete and real uses of the theoretical ideas you'll use in 455
- COMP520 – Compilers
  - Building a true programming language compiler requires techniques you'll see first here and gain far more depth on in 520. You'll also handle semantic analysis (type checking) and generate machine code in a way we will not.
- COMP530 – Operating Systems
  - You will gain foundational concepts of OS APIs in this course which will help you succeed in 530.

# How to ask questions:

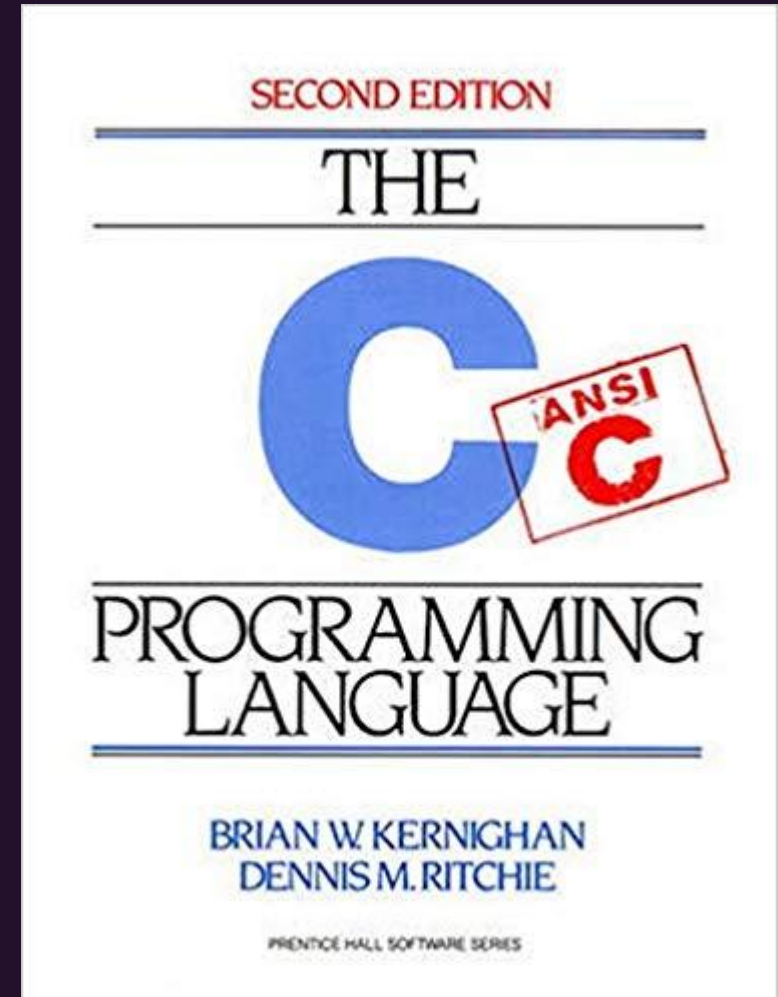
- Login to PollEverywhere: **[polleverywhere.com/auth/saml/unc](https://polleverywhere.com/auth/saml/unc)**
- Questions can be asked at: **[PolleEv.com/compunc](https://PolleEv.com/compunc)**

# Required Computing Capabilities

- Your computer **MUST** meet Carolina Computing Initiative minimums
  - Required in order to run the software we will use
- If you are on Windows...
  - ... it is *highly likely* you will need to update to the May 2020 release of Win10.
  - This process can take an hour or two depending on how out-of-date your current Windows 10 install is.
- If you are on Mac... you're probably ok!
  - Updating to the latest operating system and security updates is expected.

# Textbook #1: The C Programming Language, Second Edition

- Written by Brian Kernighan and Dennis Ritchie
- A CLASSIC book in the field
- Available at Student Stores or on Amazon
- REQUIRED readings from this text begin next week



# Text #2: Learn a Command-line Interface

- Written by yours truly...
  - ...corrections and suggestions for improvement are welcomed!
- Free!
- Only a couple of short chapters to kick us off this semester and get you oriented in a \*nix command-line environment
- Available on Sakai under Resources
- First readings assigned:
  - Getting Started - By Wednesday's class 8/12
  - Chapter 1 (10pp) - By Friday's class 8/14
  - Chapter 2 (14pp) - By next Monday's class on 8/17

# Graded Components of the Course

- Preparation and Practice - 50%
  - Programming Labs - 30%
  - Quizzes and Guided Reading Questions - 15%
  - In-class Participation - 5%
- Mastery - 50%
  - Midterms (2x) - 30%
  - Final - 20%



# Midterm and Final Schedule

- To be announced!
- If you are absent for a midterm, then your final exam will be worth 35% instead of 20%.
  - You are free to sit for a practice version of the same exam for the learning experience.
- If you take both midterms and your final exam score exceeds one of their scores, then we will retroactively count you as absent for that midterm exam.
- To pass the course, you must have a passing grade across all course components *and* take at least one midterm *and* score higher than 40% on the final exam.

# Late Assignment Policy

- No late credit for Guided Reading Questions (GRQs)
  - The lowest two GRQs will be dropped including 0s / lates
- Late programming assignments accepted up to two days late with a 20% late penalty

# How to get help?

- Office Hours via Course.Care - Enroll Code: **E533EE**
  - We will post the office hours schedule to course care soon
- We will have hours starting tomorrow for help getting started.
  - I will post these to Sakai and Course.Care as soon as they're finalized.
- We will use Piazza, at least to start the semester, to share solutions to common admin problems.
- We will assign each of you UTAs to correspond with for logistical questions or concerns in the coming couple of weeks.
  - Questions regarding programs will be redirected to office hours

How you  
might feel  
right now.




How you'll feel  
for the first  
month...





How you'll feel  
in November.

A still from the movie 'The Terminator' showing Sarah Connor (Linda Hamilton) with a bloody forehead and a man in the background.

It's a UNIX system.  
I know this.



Your love, enjoyment, and appreciation of the work you'll do this semester...



We'd love **feedback** throughout the semester.

- I welcome feedback on all aspects of the course
- Feel free to email me feedback directly or share with a UTA.
- **Please give us feedback while we have time to act on it!**
- I'll also take class wide feedback through the semester.

# How to ask questions:

- Login to PollEverywhere: **[polleverywhere.com/auth/saml/unc](https://polleverywhere.com/auth/saml/unc)**
- Questions can be asked at: **[Pollev.com/compunc](https://Pollev.com/compunc)**

# Homework

- By Wednesday - 8/12
  - [learn-a-cli-book.pdf](#) - Preqs Chapter - Get software setup
- By Friday - 8/14
  - [010-the-sorcerers-shell.pdf](#) - Introduction to a Shell
  - Take one side of one page of hand-written notes for Quiz 0
- By Monday - 8/17
  - [020-directories-files-paths.pdf](#) - Intro to File System
  - Buy Textbook: The C Programming Language 2nd Edition