

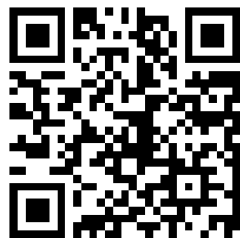
LEC 16

**CSE 123**

# Victory Lap & Next Steps

Questions during Class?  
Raise hand or send here

sli.do #cse123



BEFORE WE START

*Talk to your neighbors:*


*What was the best thing you learned  
about this quarter? Why?*

Music: [123 24su Lecture Tunes](#) 

**Instructor:** Joe Spaniac

**TAs:** Andras Eric Sahej Zach  
Daniel Nicole Trien

# Lecture Outline

- **Announcements** 
- Course Recap
  - Why 123?
  - Topics Covered
- Next Steps
  - Future Courses
  - Future Projects
- Thank you!

# Announcements

- Programming assignment 3 / R6 feedback releasing today
- Programming assignment 4 due tonight (8/14) at 11:59pm
  - No resubmission opportunity, get something submitted!
  - Make sure to look at P3 feedback before submitting
- R7/8 due Friday (8/16) at 11:59pm
  - Two forms to submit! Both on the Ed board.
  - R8 open to all assignments w/ feedback released
- IPL closes end of day Thursday (8/15), no TAs there Friday (8/16)
- **Final Exam this Friday (8/16) @ 10:50-11:50am in GWN 301**
  - Seating chart now posted on the course website!
  - Typical rules for quizzes, note sheet (8.5" x 11" double-sided, typed or handwritten)

# Lecture Outline

- Announcements
- **Course Recap** 
  - Why 123?
  - Topics Covered
- Next Steps
  - Future Courses
  - Future Projects
- Thank you!

# [Recap] Why 123?

1. To solve more complex problems by leveraging more complex programming structures / patterns
2. To better rationalize specific design decisions
  - How to “best” structure classes to reduce redundancy
  - Which ADT implementations are “most” appropriate to use
3. To understand and critically analyze intersections between Computer Science and society
  - Search engines, algorithmic art, machine learning, etc.
  - Developing informed opinions on current issues



**Be a better programmer**



**Be a better person**


# [Recap] Topics Covered

- Advanced Object-Oriented Programming (OOP)
  - Inheritance, Polymorphism, Abstract classes
- Implementing Abstract Data Types (ADTs)
  - ArrayIntList (int[] elementData, int size)
  - LinkedIntList (ListNode front)
  - Java's ArrayList & LinkedList (int size, ListNode back)
- Runtime (Complexity & Big O notation)
- Recursion
  - Recursive definitions ( $n! = n * (n - 1)!$ )
  - (Implicit) Base and Recursive cases
  - Public / private pairs
  - LinkedLists w/ recursion ( $x = \text{change}(x)$ )
- Binary Trees
  - Binary Search Trees (BST) & Runtime
- Exhaustive Search / Recursive Backtracking
  - Dead ends / Choose, explore Un-choose
- Machine Learning & Hashing

Assessable  
content

**You've  
learned  
A LOT!!!**  
*(hopefully)*

# Lecture Outline

- Announcements
- Course Recap
  - Why 123?
  - Topics Covered
- **Next Steps** 
  - Future Courses
  - Future Projects
- Thank you!

# Future Courses

- Tons of options for everyone!
  - Self study always valid too!

## CSE Majors

Course	Overview
CSE 311	Mathematical foundations
CSE 351	Low-level computer organization/abstraction
CSE 331	Software design/implementation
CSE 341	Programming languages
CSE 344	Data Management (databases)

<https://www.cs.washington.edu/academics/ugrad/current-students>

## Non-CSE Majors

Course	Overview
CSE 154	Intro to web programming
CSE 163	Intermediate programming, data analysis
CSE 180	Introduction to data science
CSE 373	Data structures and algorithms
CSE 374	Low-level programming and tools
CSE 412	Data Visualization
CSE 416	Intro. to Machine Learning

<https://www.cs.washington.edu/academics/ugrad/nonmajor-options/nonmajor-courses>



# Future Projects


- At this point, you know 90% of the fundamentals you need to accomplish practically any project
  - Hurdle will typically be learning the syntax of a new language, using github, importing external libraries, etc.
- Some ideas:
  - Make a Minecraft mod! (Java) [[link](#)]
  - Make discord bot! (Python) [[link](#)]
  - Make personal website! (HTML, CSS, Javascript) [[link](#)]
  - Convert a project from this course into a more user-friendly application
    - C2, make a Graphical User Interface (GUI) [[link](#)]
    - P4, refine the Email class until you get an accuracy you're happy with
  - Really, anything you want!!!\*

\*Warning: it's often hard to tell what computers can do easily and what they struggle with...

# Frequently Asked Questions

- How can I get better at programming?
  - Practice! Check out [leetcode](#) for a *ton* of practice problems
- How can I learn to X?
  - Search online, read books, look at examples
  - Start with something that already works (try [github](#)), then make changes!
- What should I work on next?
  - Anything you can think of! (See previous slide for some ideas)
- Should I learn another language? Which one?
  - Depends on what you want to do!
    - Python: Data Science & Machine Learning
    - JavaScript: Web Dev
    - C / C++: Systems Programming

# Lecture Outline

- Announcements
- Course Recap
  - Why 123?
  - Topics Covered
- Next Steps
  - Future Courses
  - Future Projects
- **Thank you!** 

# Thank You!

- Thank you for participating, asking questions, engaging with course materials & resources!
  - Feedback if you filled out the course evaluation :)
- Your amazing TAs!



- Any final questions before we wrap?