LEC 01

CSE 373

Welcome!

BEFORE WE START

***Use the Zoom chat*:**
Introduce yourself! What are you most excited about in this class? What's your least favorite vegetable?

Music: Carly Rae Jepsen

Instructor	Hunter Schafer	
TAs	Ken Aragon	Paul Pham
	Khushi Chaudhari	Mitchell Szeto
	Joyce Elauria	Batina Shikhalieva
	Santino Iannone	Ryan Siu
	Leona Kazi	Elena Spasova
	Nathan Lipiarski	Alex Teng
	Sam Long	Blarry Wang
	Amanda Park	Aileen Zeng

Lecture Outline

- Introductions 
- About this Course
  - Course Components & Tools
  - Policies
  - Making the Most of this Class
- Abstract Data Types

Course Staff

- Instructor: Hunter Schafer
  - Assistant Teaching Professor in the Allen School
  - Used to be a student at UW just like you!

- Teaching Assistants:

  - Available in section, office hours, discussion board, and 1:1 meetings
  - Invaluable source of information & help in this course

- We're excited to get to know you!
  - Our goal is to help you succeed

Students

- Currently 240 students registered for the course

- Strength in numbers
  - With 240 students, if you're confused about something, I guarantee someone else is too!
  - Students come from all different backgrounds & majors

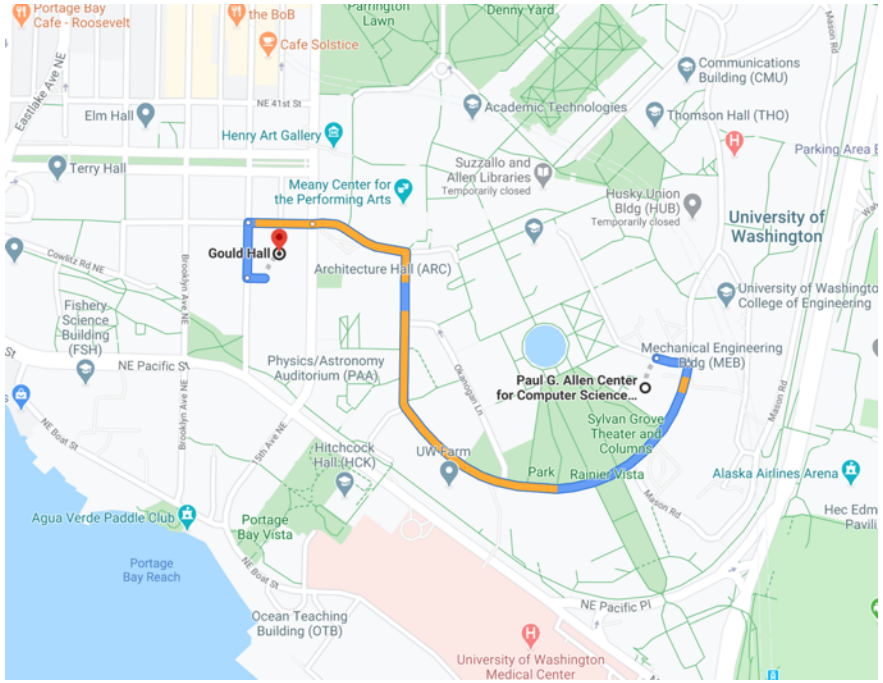
- Focus on us trying to help you build community
  - Meet others in the class to form study groups or have people you can work with.

What is this Class?

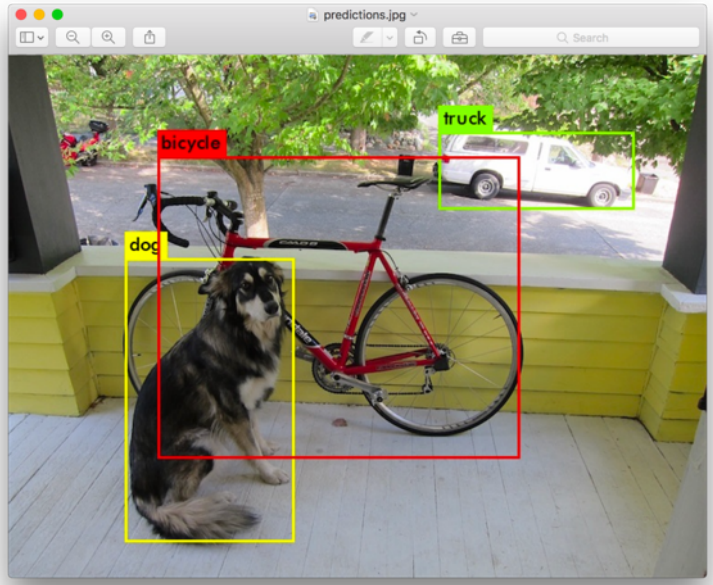
- **CSE 143 – Object Oriented Programming**
  - Classes and Interfaces
  - Methods, variables and conditionals
  - Loops and recursion
  - Linked lists and binary trees
  - Sorting and Searching
  - O(n) analysis
  - Generics
- **CSE 373 – Data Structures and Algorithms**
  - Design decisions
  - Design analysis
  - Implementations of data structures
  - Debugging and testing
  - Abstract Data Types
  - Code Modeling
  - Complexity Analysis
  - Software Engineering Practices

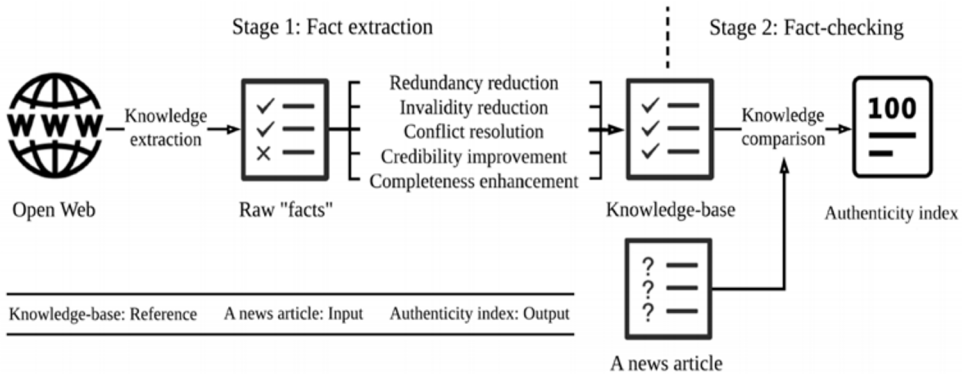
Why 373?

1. Build a strong foundation of data structures and algorithms that will let you tackle the biggest problems in computing



373 Data Structures & Algorithms





Fake News: A Survey of Research, Detection Methods, and Opportunities (Xinyi Zhou, Reza Zafarani/arXiv:1812.00315)

Why 373?

2. Pick up the vocabulary, skills, and practice needed to make **design decisions**. Learn to **evaluate** the tools in your CS toolbox

- Differences between technical implementations
- Evaluation can mean many different things!

Lecture Outline

- Introductions
- **About this Course**
  - **Course Components & Tools**
  - Policies
  - Making the Most of this Class
- Abstract Data Types

Course Components

LECTURES (x29)

- Delivered async. via ItemPool
- Live class sessions for interactive practice

SECTIONS (x10)

- Held live via Zoom
- More practice, reviews, applications
- TA advice, how to be an effective student
- Preparation for exams

PROJECTS (x5)

- Partner recommended
- Programming in Java
- Applying & implementing course concepts
- More practical

EXERCISES (x5)

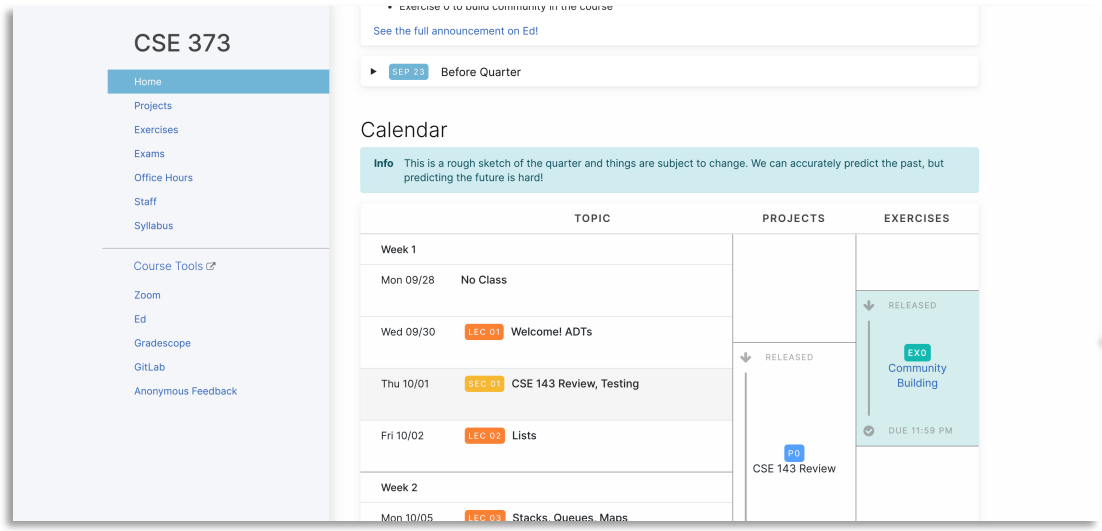
- Individual
- Written problems, focusing on the "why"
- More conceptual

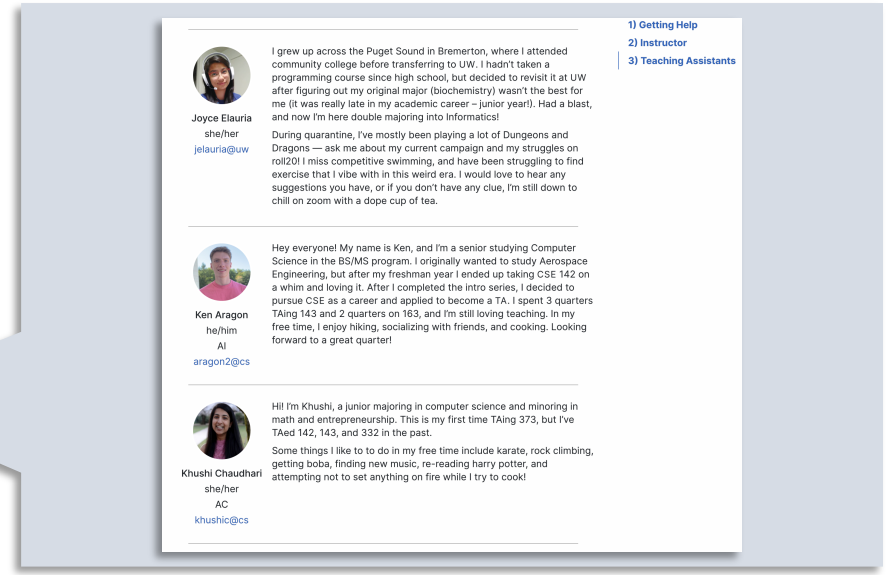
EXAMS (x2)

- Available over a multi-day window, complete whenever works for you
- More details to come

Course Website

cs.uw.edu/373





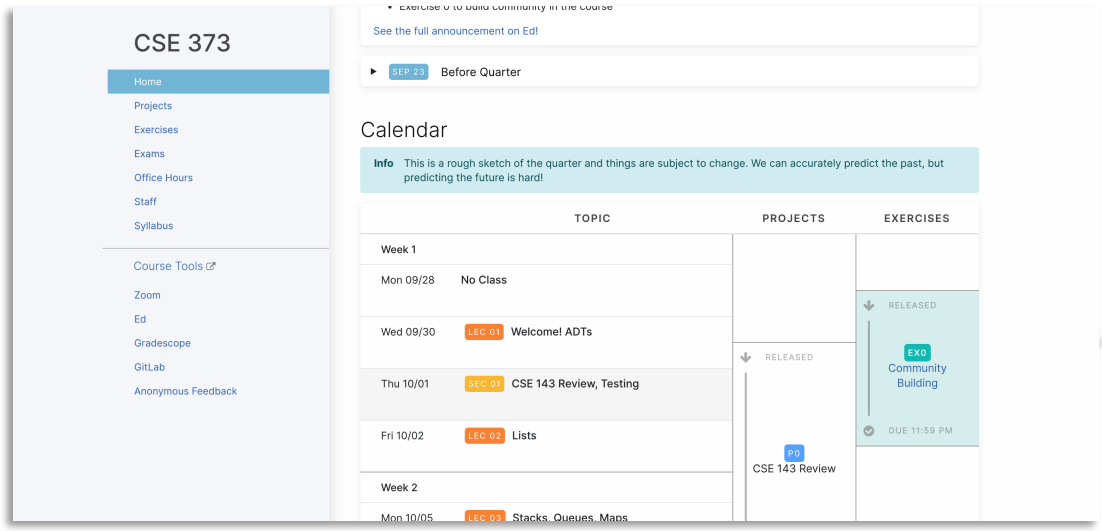
Get to know the staff

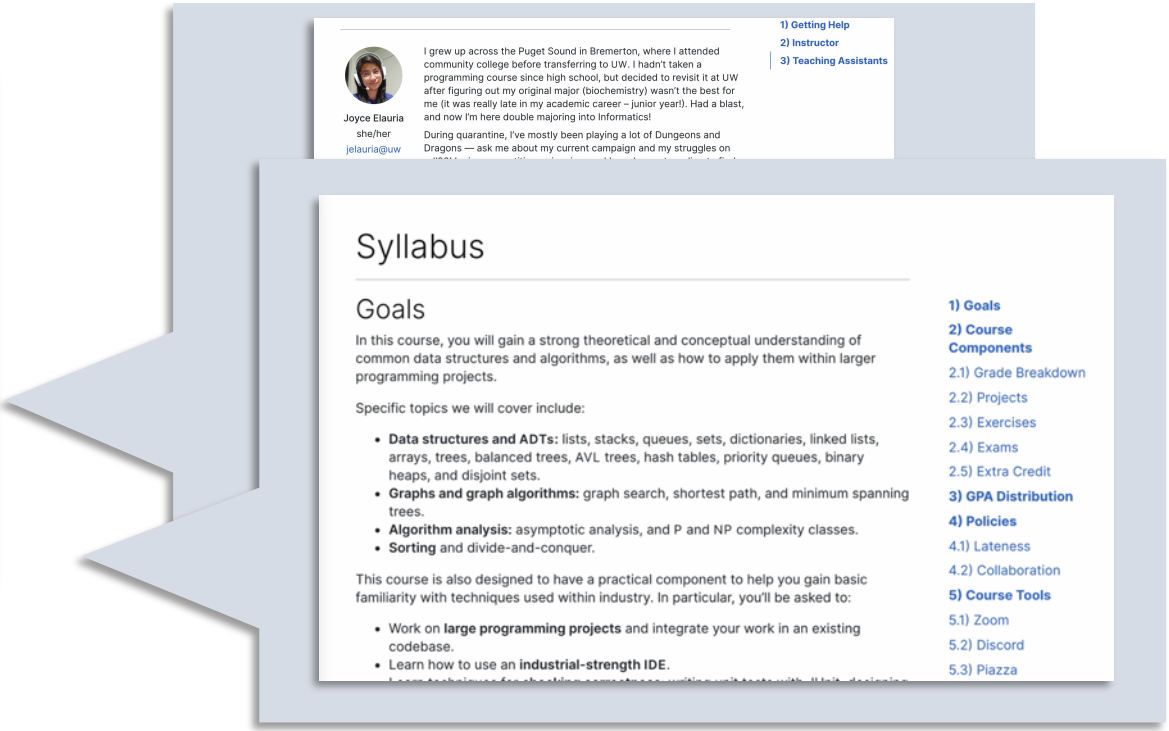
Contains most course info – check frequently!

- Announcements, Calendar, Lecture Slides, Assignment Specs, Office Hours schedule, Staff Bios, Important Links

Course Website

cs.uw.edu/373





Contains most course info – check frequently!

- Announcements, Calendar, Lecture Slides, Assignment Specs, Office Hours schedule, Staff Bios, Important Links

Please familiarize yourself with the course syllabus this week!

Other Course Tools

Ed
- Discussion Board & Announcements
- Please ask AND answer!
- Anonymous option

Discord
- Community: meet other students, form study groups
- Office Hours queue here
- *More details on website*

Gitlab
- Everyone gets a git repo
- We'll distribute starter code, you'll push your work
- *More details to come*

Gradescope
- Submit all your assignments
- Get feedback

Canvas
- Only used for Zoom recordings and gradebook

Lecture Outline

- Introductions
- **About this Course**
  - Course Components & Tools
  - **Policies**
  - Making the Most of this Class
- Abstract Data Types

Grading Breakdown

- Your grade will consist of the following weighted categories:

Category	Weight
Programming Projects	45%
Individual Exercises	25%
Exam I	15%
Exam II	15%

- Instead of curving the class as usual, we'll use a bucket system:
  - These are *minimum* GPA guarantees – may adjust upward

Percentage	Course Grade
90%	3.5
80%	3.0
70%	2.5
60%	2.0
50%	0.7

Assignment Policies

Collaboration & Academic Integrity

- These concepts are hard: we strongly encourage discussion + collaboration!
  - Don't attempt to gain credit for something you didn't do
  - In general, share ideas and work together, but don't copy work. Never show someone else your code or solution write up.
  - Always cite the help you receive
  - Full collaboration with your partner on projects!
- **Read full policy in Syllabus**

Lateness

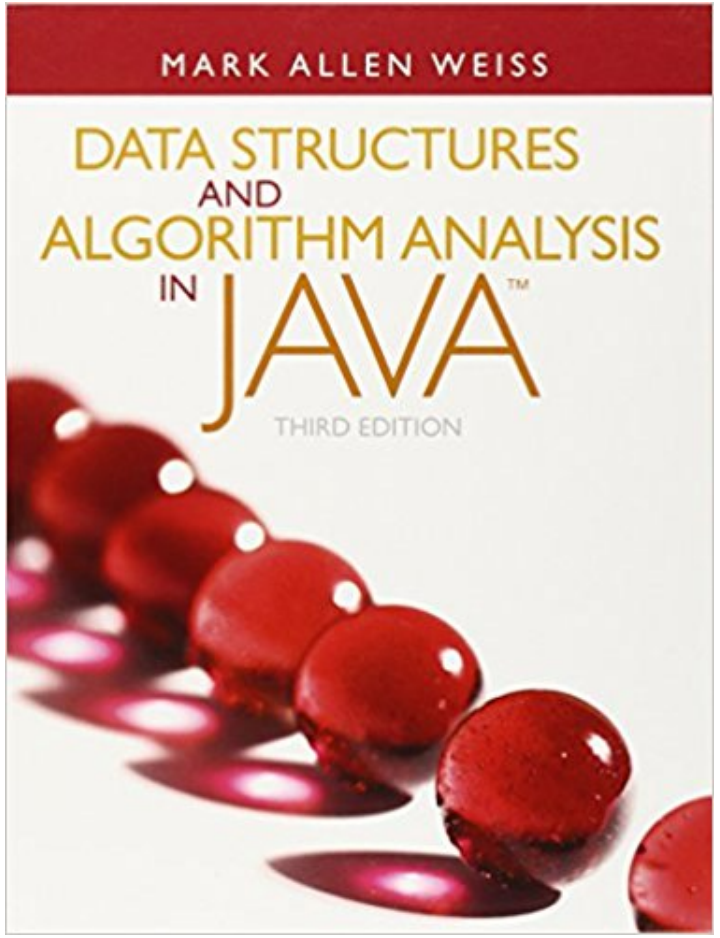
- You get 7 "free" late days for the quarter – submit 24 hours late with no penalty
  - Use on projects or exercises
- After that, -5% each day late
- No assignment can be submitted >72 hours late
  - Except with instructor permission

Textbook

- Data Structures and Algorithm Analysis in Java by Mark Allen Weiss

- Completely **optional**
  - Nothing assigned out of the textbook
  - No readings

- Advice: only purchase if you learn best with a textbook, otherwise not recommended



Lecture Outline

- Introductions
- **About this Course**
  - Course Components & Tools
  - Policies
  - **Making the Most of this Class**
- Abstract Data Types

Getting Help

- Discussion Board
  - Feel free to make a public or private post on Ed
  - We encourage you to answer other peoples' questions! A great way to learn
- Office Hours
  - TAs can help you face to face in office hours, and look at your code
  - Discord gives great flexibility – feel free to join your peers in the chats to discuss
- Section
  - Work through related problems, get to know your TA who is here to support you
- Your Peers
  - We encourage you to form study groups! Discord or Ed are great places to do that
- Email
  - We prefer that all content and logistic questions go on the Ed discussion board (even if you make them private). 240 of you >>> 17 of us!
  - For serious personal circumstances, you can email me directly. It never hurts to email me, but if it's a common logistic question, I will politely tell you to post on the discussion board.

Help Us Improve!

- We're still learning how to do this online ☺
  - Thank you in advance for your patience and understanding
  - We *really* value your feedback!
  - Let us know what's working and what isn't working for you
  - Something that went well in another course? Tell us about it!

- Post on the discussion board (can be public/private).
  - Note: Anonymous here is anonymous to other students, not to the staff.

- Submit feedback via the **Anonymous Feedback Tool** (linked under "Course Tools" on the website)

Metacognition

- **Metacognition**: asking questions about your solution process.

- Examples:
  - **While debugging**: explain to yourself why you're making this change to your program.
  - **Before running your program**: make an explicit prediction of what you expect to see.
  - **When coding**: be aware when you're not making progress, so you can take a break or try a different strategy.
  - **When designing**:
    - Explain the tradeoffs with using a different data structure or algorithm.
    - If one or more requirements change, how would the solution change as a result?
    - Reflect on how you ruled out alternative ideas along the way to a solution.
  - **When studying**: what is the relationship of this topic to other ideas in the course?

The World Around 373

- Our goal is to give you a great 373 experience
  - But CSE 373 does not exist in a vacuum – there's a lot going on in the world right now that can impact your education
- We've designed course policies for maximum flexibility: plenty of late days, take-home exams, no participation
  - But we cannot cover every individual situation

- **Please reach out** if you need accommodations of any kind to deal with these unfamiliar situations

Lecture Outline

- Introductions
- About this Course
  - Course Components & Tools
  - Policies
  - Making the Most of this Class
- **Abstract Data Types** 

Data Structures & Algorithms

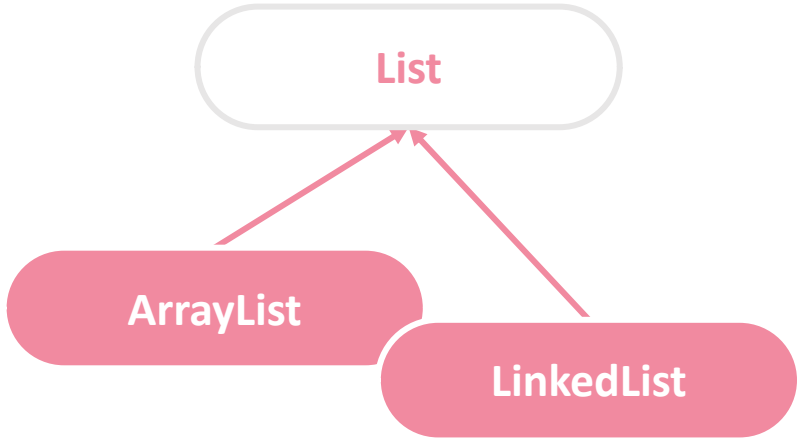
- Data Structure:
  - A way of organizing, storing, accessing, and updating data
  - **Examples from CSE 143**: Arrays, Linked Lists, Stacks, Queues, Trees

- Algorithm:
  - A series of precise instructions to produce a specific outcome
  - **Examples from CSE 14**: Binary Search, Merge Sort, Recursive Backtracking

Review Interface vs. Implementation

- In Java, an **interface** is a data type that specifies what to do but not how to do it.
  - **List**: an ordered sequence of elements.

- A **subtype** implements all methods required by the interface.
  - **ArrayList**: Resizable array implementation of the List interface.
  - **LinkedList**: Doubly-linked implementation of the List interface.

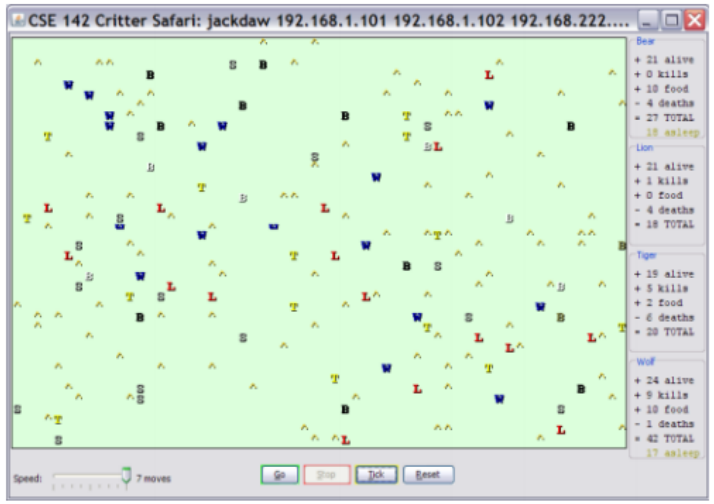


Review Client vs. Object

Client Classes

- A class that is executable, in Java: contains a Main method

```
public static void main(String[] args)
```



Object Classes

- A coded structure that contains data and behavior
- Start with the data you want to hold, organize the things you want to enable users to do with that data

1. Ant

constructor	`public Ant(boolean walkSouth)`
color	red
eating behavior	always returns `true`
fighting behavior	always scratch
movement	if the Ant was constructed with a `walkSouth` value of `true`, then alternates between south and east in a zigzag (S, E, S, E, ...); otherwise, if the Ant was constructed with a `walkSouth` value of `false`, then alternates between north and east in a zigzag (N, E, N, E, ...)
toString	`"%"` (percent)

ADTs: Abstract Data Types

- Java interfaces represent the concept of abstract data types.
- An **abstract data type** is a data type that does not specify any one implementation.
- **Data structures** implement ADTs.
  - **Resizable array** can implement List, Stack, Queue, Deque, PQ, etc.
  - **Linked nodes** can implement List, Stack, Queue, Deque, PQ, etc.

List ADT

A collection storing an ordered sequence of elements.

- Each element is accessible by a zero-based index.
- A list has a size defined as the number of elements in the list.
- Elements can be added to the front, back, or any index in the list.
- Optionally, elements can be removed.

Where we're Headed: ADTs we'll look at

- List
- Set
- Map
- Stack
- Queue
- Priority Queue
- Graph
- Disjoint Set

Learning to Bake in a CSE Class

- Think of what you'll learn this quarter as a cookbook
  - ADTs are the chapters/category: Soups, Salads, Cookies, Cakes, etc
    - High-level descriptions of a category of functionality
    - You don't serve a soup when guests expect a cookie!
  - Data structures are the recipes: chocolate chip cookies, snickerdoodles, etc
    - Step-by-step, concrete descriptions of an item with specific characteristics
    - Understand your tradeoffs before replacing carrot cake with a wedding cake
- When you go out into the world …
  - Figure out which category is required
  - Choose the specific recipe that best fit the situation