

LEC 00

CSE 122

Welcome!

Questions during Class?

Raise hand or send here

sli.do #cse122



## BEFORE WE START

***Talk to your neighbors:***  
*Introduce yourself to your neighbor!*

*What is your name? Major? What did you do over Spring break?*

---

**Instructors** **Tristan Huber & Hunter Schafer****TAs**

Ambika  
Andrew  
Audrey  
Autumn  
Ayush  
Ben  
Colton  
Di  
Eesha  
Elizabeth

Evelyn  
Jacob  
Jaylyn  
Jin  
Joe  
Kevin  
Leon  
Megana  
Melissa  
Mia

Poojitha  
Rishi  
Rucha  
Shivani  
Shreya  
Steven  
Suhani  
Yijia  
Ziao

# Lecture Outline

- **Introductions** 
- About this Course
  - Course Components & Tools
  - Policies
  - Making the Most of this Class
- Intro/Review Java

# Course Staff

- Instructors: Tristan Huber & Hunter Schafer
- Teaching Assistants: [27 Awesome TAs](#)
  - Available in section, office hours, and discussion board
  - Invaluable source of information & help in this course
- We're excited to get to know you!



Tristan



Hunter

# Students

- Currently ~400 students registered for the course!
- Strength in numbers
  - With ~400 students, if you're confused about something, I guarantee someone else is too!
  - Students come from all different backgrounds & majors & interests in future career goals.
- 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 121 – Computer Programming I or Other Programming Experience

- Print statements
- Data types (int, String, boolean)
- Methods / Functions
  - Parameters
  - Returns
- Control structures
  - Loops
  - Conditionals
- File I/O
- Arrays
- **Computational Thinking**  
(language agnostic)

## CSE 122 – Computer Programming II

- Decomposing large problems into smaller, manageable, subproblems
- Using data structures
  - List
  - Stacks / Queues
  - Sets
  - Maps
- Object Oriented Programming
  - Interfaces

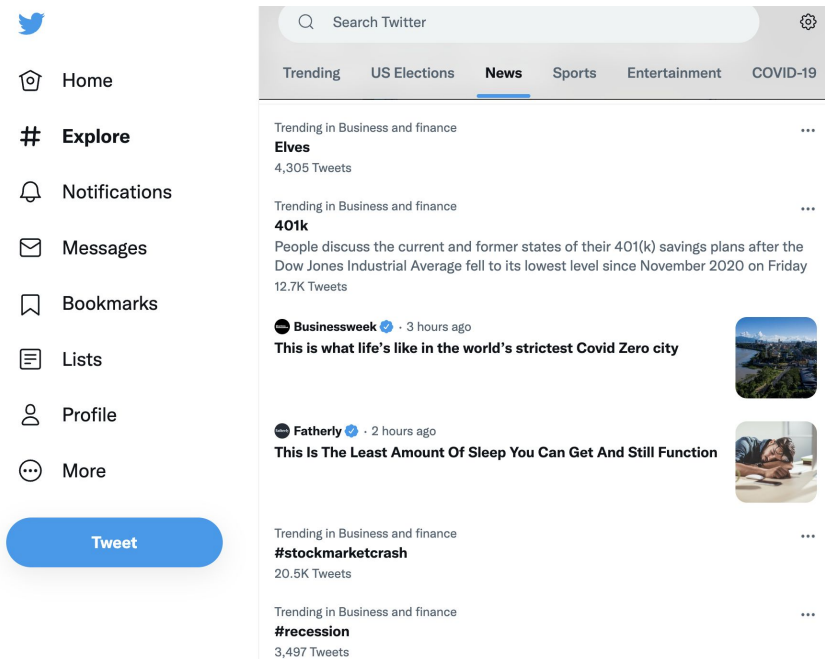


# Prerequisite Knowledge

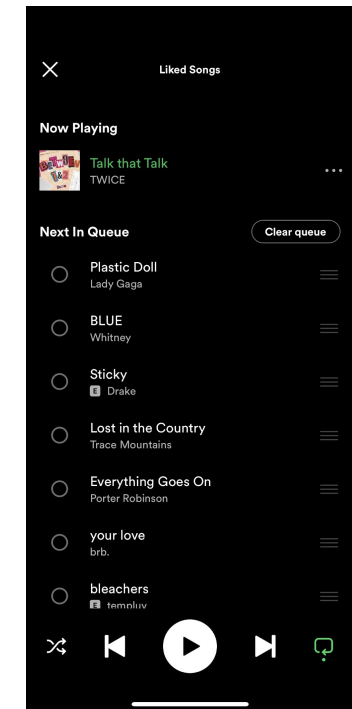
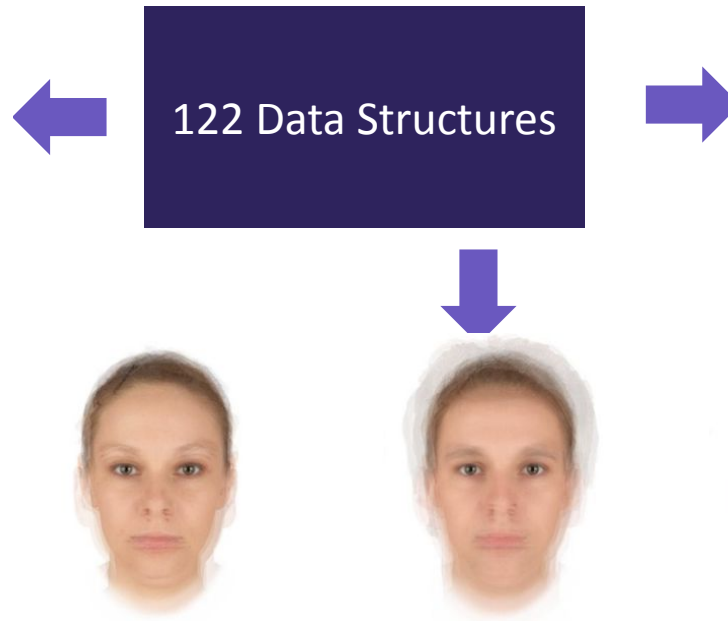
- Students entering CSE 122 are coming from many of different backgrounds
  - UW: CSE 121 or other intro programming course
  - Community College: Intro Programming Course
  - High School Programming Course (e.g., UWHS, AP CS, IB CS, etc.)
  - Self-taught or other previous experience
- Importantly: CSE 122 is in Java, but we **do not expect prior experience in Java!** Do expect knowing the list of CSE 121 topics in some language.
  - Students who do not have experience in Java will be focusing on practicing the programming skills you know in a new language!
  - You will find the [Java Tutorial](#) and Programming Assignment 0 very helpful!
- If you want to know if this class is the right fit for you, take the [Allen School Self-Placement Test](#)

# Why 122?

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



Source: Twitter 9/23



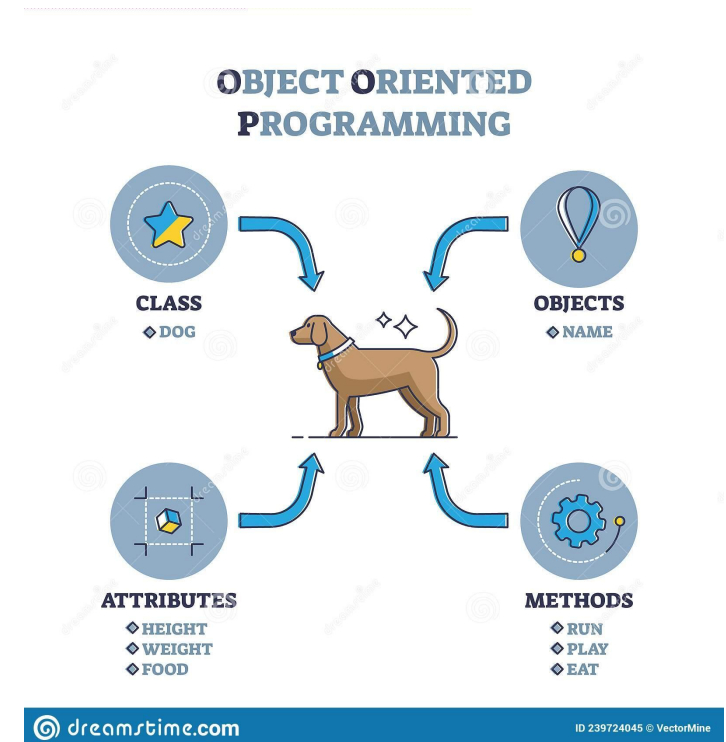
Source: Hunter's Spotify

Source: Ethical CS

# Why 122?

2. Learn an important structural pattern for representing **objects** in code to make our code more **reusable** and **maintainable** and **easier to understand**.

- Java is designed around this idea of objects. We haven't been leveraging that yet!
- Used in almost every real-world software project.





# Lecture Outline

- Introductions
- **About this Course**
  - **Course Components & Tools** 
  - Policies
  - Making the Most of this Class
- Intro/Review Java

# Course Components

## Meetings

### LECTURES

(x20)

- We're here!
- Introduce concepts, practice ideas, discuss applications.
- Pre-class materials to prepare for class each day. Due **before** class.

### SECTIONS

(x19)

- Held in person
- More practice, reviews, applications
- TA advice, how to be an effective student
- Preparation for quizzes / exams

## Assessments

### PROGRAMMING ASSIGNMENTS

(x4)

- Structured assignments
- Programming in Java
- Applying & implementing course concepts

### CREATIVE PROJECTS

(x4)

- More open-ended assignments
- Explore new ideas and applications

### QUIZZES

(x3)

- Taken in quiz section
- 45 minutes on computer
- One retake per quiz

### EXAM

(x1)

- Culminating exam
- **Tuesday 6/6 @ 2:30 pm**

# Course Website

[cs.uw.edu/122](https://cs.uw.edu/122)

The screenshot shows the CSE 122 course website. On the left is a navigation sidebar with links: Home / Calendar, Programming Assignments, Creative Projects, Exam, Staff, Office Hours, Syllabus, Grading Rubric, COVID-19 Safety, Resources, Course Tools (external), EdStem, and Anonymous Feedback. The main content area is titled 'Introduction to Computer Programming II Spring 2023'. It features a yellow registration notice, a blue feedback notice, and a welcome message. Below the welcome message are two expandable sections: 'What is this class? What will I learn?' and 'Prior Experience and Expectations'. At the bottom, there's a 'This Week (at a glance)' section showing a class session on Wednesday (03/29) and a quiz section on Thursday (03/30).

CSE 122

Home / Calendar

Programming Assignments

Creative Projects

Exam

Staff

Office Hours

Syllabus

Grading Rubric

COVID-19 Safety

Resources

Course Tools ↗

EdStem

Anonymous Feedback

## Introduction to Computer Programming II

### Spring 2023

**Registration** Do not email the course staff or instructor requesting an add-code for the course. The course staff do not have any add-codes. Please email [ugrad-advisor@cs.washington.edu](mailto:ugrad-advisor@cs.washington.edu).

**Feedback** Feedback is always welcome! You can contact the the course staff or submit anonymous feedback.

### Welcome to CSE 122: Introduction to Computer Programming II 🎉

► What is this class? What will I learn?

► Prior Experience and Expectations

If you want to learn more about the course and its policies, please check out our [course syllabus](#).

### Announcements

### This Week (at a glance)

**Wednesday (03/29)**

- 📅 Class Session @ 2:30pm in KNE 120

**Thursday (03/30)**

- 📅 Quiz Section 0: Welcome

## Instructor



**Hunter Schafer** HEMMANS  
[hschafer@cs](mailto:hschafer@cs)

Hi there! 🌟 My name is Hunter! I've lived in Seattle for almost my whole life after a brief stint in Alaska, where I was born, and a couple of years in Minnesota. I did my undergrad and masters here at UW in the Paul G. Allen School of Computer Science & Engineering, and now, I work here full-time on the Allen School faculty!

**Office Hours**  
Mondays  
2:00-4:00pm  
CSE 530 or Zoom

Funnily enough, I had absolutely no idea what computer science was when I started at UW. I took CSE 142 on a whim and enjoyed it enough to continue on to CSE 143, but not enough to commit to it as a field of study. It wasn't until I became a teaching assistant for the 14x series did I really find my passion for computer science: teaching computer science. Since then, I've shaped my college and post-college career around teaching computer science. Since graduating with my masters, I've worked here at the Allen School as a Teaching Professor.

Outside of school and work, I usually spend most of my time hanging out with friends: just enjoying quality time or going out and exploring new restaurants and coffee shops. While living through the inside-times, I've tried to make a routine with some of my available free time: I've gotten into reading and cooking more regularly, I started practicing yoga, and I even tried my hand at streaming on Twitch (all while drinking questionably unhealthy amounts of coffee).

Talk with me about your favorite coffee shops or restaurants around Seattle, I'm always looking for new places to check out! Good book, podcast, or game recommendations are great too!



**Tristan Huber** HEMMANS  
[hubert4@cs](mailto:hubert4@cs)

Hi, I'm Tristan!

I grew up in Seattle and have lived here most of my life, excepting a few years spent in New York City.

Contains most course info – check frequently!

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

Get to know the staff

# Course Website

[cs.uw.edu/122](https://cs.uw.edu/122)

## Calendar

**Info** This is a rough sketch of the quarter and things are subject to change. We can accurately predict the past, but predicting the future is hard!

**Lessons** Anything listed in the "Lesson" materials for a day should be read **before** attending class that day. The Lessons are a first introduction to the most important terms and concepts for that day of class. It is okay if the Lesson doesn't make complete sense as we have the rest of the class day to clarify the concepts, but if you don't do the Lesson the class session won't make any sense.

[Jump to Today](#) [Expand all Below](#)

Topic	Programming / Creative Projects	Resubmissions
Module 0 - Welcome, Functional Decomposition, Design		
Tue 01/03	No section today!	
Wed 01/04	<b>LES 00</b> Welcome; Syllabus Details <i>Note: Normally you would complete the Pre-class Work before class. There is nothing you need to complete before class today!</i>	
Thu 01/05	<b>SEC 00</b> Welcome	
Fri 01/06	<b>LES 01</b> Java Review/Introduction; Functional Decomposition	

Contains most course info – check frequently!

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

**CSE 122**

Home / Calendar  
Programming Assignments  
Creative Projects  
Exam  
Staff  
Office Hours  
**Syllabus**  
Grading Rubric  
COVID-19 Safety  
Resources  
Course Tools  
EdStem  
Anonymous Feedback

## Syllabus

### Course Information

**Teaching Staff**  
Instructor: Miya Natsuhara  
Instructor Email: mnats@cs.washington.edu  
Registration Questions: CSE Advisors (ugrad-advisor@cs.washington.edu)  
Course Staff and Support Hours: Course Staff and Office Hours

Who to contact?

### Class Session Meeting

See Class Sessions for information on how each day of class will be run.

- WF: 11:30 am - 12:20 pm (KNE 130)
- WF: 2:30 pm - 3:20 pm (KNE 120)

- 1) Course Information
- 2) Course Goals
- 2.1) Learning Objectives
- 3) Software and Textbooks
- 4) Class Sessions and Quiz Sections
- 4.1) Class Sessions
- 4.2) Quiz Sections
- 5) Inclusion
- 6) Required Course Work, Resubmissions, and Late Work
- 7) Getting Help from Staff & Peers
- 8) Course Climate
- R.1) Extenuating

**Please familiarize yourself with the course syllabus this week!**

# Other Course Tools



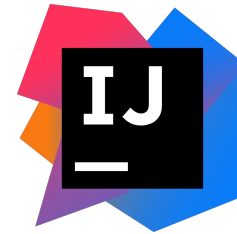
## Ed

- Community & Information
  - Discussion Board  
(please ask & answer!; anonymous option)
  - Chat
  - Announcements
- Pre-Class Materials / Section Handouts
- Assignments
  - Online IDE
  - Submit assignments
  - View Feedback

My Digital Hand

## My Digital Hand

- Queueing in office hours



## IntelliJ

- Develop offline
- Visual debugger



## Canvas


- Gradebook
- Lecture recordings



## Sli.do

- In-class activities  
(ungraded)
- No account needed

# Lecture Outline

- Introductions
- **About this Course**
  - Course Components & Tools
  - **Policies** 
  - Making the Most of this Class
- Intro/Review Java



# Resubmissions / Retakes

*Learning is a challenging process that takes time, it doesn't always happen on your first try.*

- Each week, one previous Programming Assignment or Creative Project can be resubmitted
  - Must be accompanied by write up explaining changes
  - Grade on resubmission replaces original grade.
  - *Tip: Resubmit as early as possible*
- Each quiz can be retaken at most once

See syllabus for more details

# Collaboration

- These concepts are challenging: 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.
  - For any ungraded work (e.g., pre-class materials) there is no concern about academic misconduct! You should be collaborating on those without reservation.
  - On graded assignments you should still collaborate, but the code you write should be of your own creation.
  - Always cite the help you receive on graded work
- [Withdrawal Policy](#)
- **Read full policy in Syllabus**

# Textbook

## Pre-class Materials

- All required readings are available free on Ed!
- Should be finished before class (not graded)

## Optional Textbook

- [Building Java Programs by Reges and Stepp \(5<sup>th</sup> Edition\)](#)
- Not required but does add another perspective. Will reference relevant chapters.
- Advice: only purchase if you learn best with a textbook, otherwise not recommended.

ed CSE 122 - 22au - Ed Lessons

< Lessons Slides Prev Next

Arrays Review

[Pre-Class Work] ArrayLists

Arrays Review ✓

ArrayList Basics

ArrayList Methods

Syntax: Arrays vs. ArrayList

ArrayLists [Video Walkthrough]

ArrayList Review

100% ArrayList Programming Review

Count Unique

### Arrays Review

Previously in CSE 121, we had learned about **arrays** – a data structure that can hold multiple types!

As mentioned previously, we like to think of arrays as **cubbies** – or a group of variables that one data structure. Remember that arrays have the following (with an accompanying diagram):

1. a **name**
2. a **specific length** (number of compartments)
3. a **specific type** that each of its compartments can hold
4. compartments where each compartment has:
  - an **index** (like `String` indices, starting at index 0)
  - the ability to hold a piece of data

Remember to initialize an array, you need the following:

1. **type[]** – start by listing the type of your array and its elements and make sure to have closing square brackets to signify this is an array.
  - 1. Examples: `String[]`, `int[]`, `char[]`, etc.
2. **name** – the name of your array can be anything, as long as it's concise, descriptive, and naming guidelines.
3. **array construction code** – the remaining code to construct a new array follows the type `length`; where the type should match the type listed on the left hand side of the line

```
int[] arr = new int[4];
```

name: `arr` (`int[]`)      0      1

# Lecture Outline

- Introductions
- **About this Course**
  - Course Components & Tools
  - Policies
  - **Making the Most of this Class** 
- Intro/Review Java

# How Learning Works

- Learning requires **active participation** in the process. It's not as simple as sitting and listening to someone talk at you.
  - Requires **deliberate practice** in **learning by doing**
  - Benefits from **collaborative learning**
- Hybrid classroom model
  - Asks you to do some preparation before class in the form of readings and practice problems.
    - Should take ~30 minutes a day
  - Class will start with brief recap, then pick up where the reading and practice problems leave off.
  - Attendance isn't graded, but showing up and trying is the first step in succeeding in the class!
- Pre-class materials are ungraded, but
  - It's okay if you find them challenging! That means you are learning!



# 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?



# 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
- Introductory Programming Lab (Office Hours)
  - TAs can help you face to face in office hours, and look at your code
  - You can go to the IPL with **any** course questions, not just assignments
- 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).
  - For serious personal circumstances, you can email Tristan and/or Hunter directly. It never hurts to email us, but if it's a common logistic question, we may politely ask you to post on the discussion board.

# Help Us Improve!

- This is a very new course! We are always looking for feedback on how to improve the class for you and for future students! Thank you in advance for your patience and understanding as we develop everything. 😊
  - 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)

# The World Around CSE 122

- Our goal is to give you a great CSE 122 experience
  - But CSE 122 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: ability to resubmit assignments and retake quizzes
  - 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
- **Intro/Review Java** 

# Hello World

- Java Specifics

- Every program needs a **class**
- Runnable programs need a **main** method (*signature* must exactly match)
- System.out.println to print
- **"Hello world"** is a String

- Running on Ed

- **Run** runs your program
- **Mark** submits and runs autograder
  - Submit as many times as you like
  - "Shotgun submission" = Unhelpful habit
- **Solution** shows solution (if applicable)

```
public class HelloDemo {  
    public static void main(String[] args) {  
        System.out.println("Hello world");  
    }  
}
```

# Review Java Syntax

[Java Tutorial](#) reviews all the relevant programming features you should familiar with (even if you don't know them in Java).

- Printing and comments
- Variables, types, expressions
- Conditionals (if/else if/ else)
- Loops (for and while)
- Strings
- Methods
- File I/O
- Arrays



# “Homework” for Next Time

- First assignment will be released Friday, but there are some things to do in the meantime.
- TODO this week
  - [Fill out the introductory survey](#)
  - Go meet your TA and classmates in Thursday’s quiz section
  - ★ Complete the pre-class material for Friday (see calendar)
  - [Check over syllabus details](#)