

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 summer break?*

Music: [122 24au Lecture Tunes](#) 

**Instructors:** Miya Natsuhara and Elba Garza

<b>TAs:</b>	Ayush	Heon	Harshitha	Aishah
	Andrew	Izak	Marcus	Ben
	Logan	Colin	Carson	Ivory
	Kyle	Jessica	Jack	Cady
	Maggie	Shivani	Connor	Diya
	Nicole H	Ken	Cora	Katharine
	Caleb	Mia	Hannah	
	Nicole W	Ashley	Leo	
	Jacob	Chaafen	Anyia	

# Lecture Outline

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

# Course Staff

- Instructors: Miya Natsuhara and Elba Garza
- Teaching Assistants: [33 Fantastic 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!
  - Our goal is to help you succeed 😊



# Students

- Currently 561 students registered for the course!
- Strength in numbers
  - With 561 students, if you're confused about something, we guarantee someone else is too! Ask questions in Slido or in class 😊
  - 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.



# CSE 12x Behavioral Expectations



# 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
- Arrays & 2D arrays
- **Computational Thinking**  
(language agnostic)

## CSE 122 – Computer Programming II

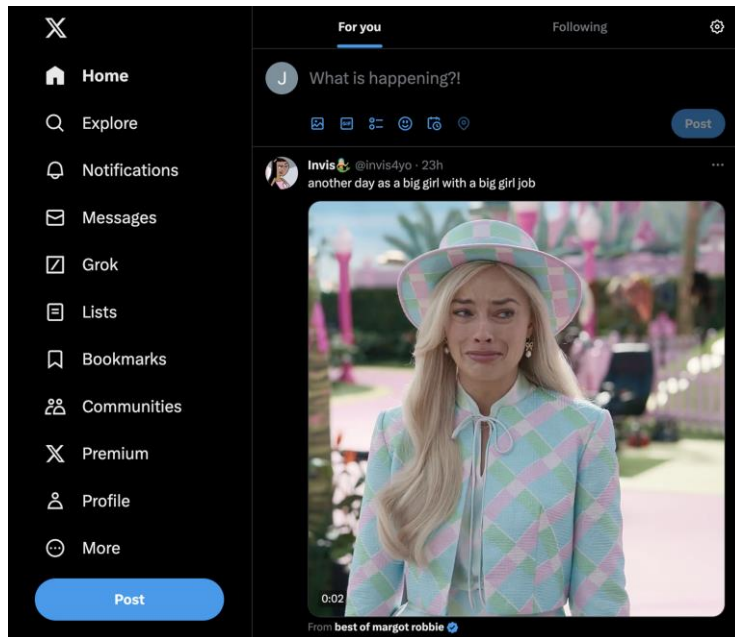
- Decomposing large problems into smaller, manageable, subproblems
- File I/O
- Using data structures
  - List
  - Stacks / Queues
  - Sets
  - Maps
- Object Oriented Programming
  - Interfaces

# Prerequisite Knowledge

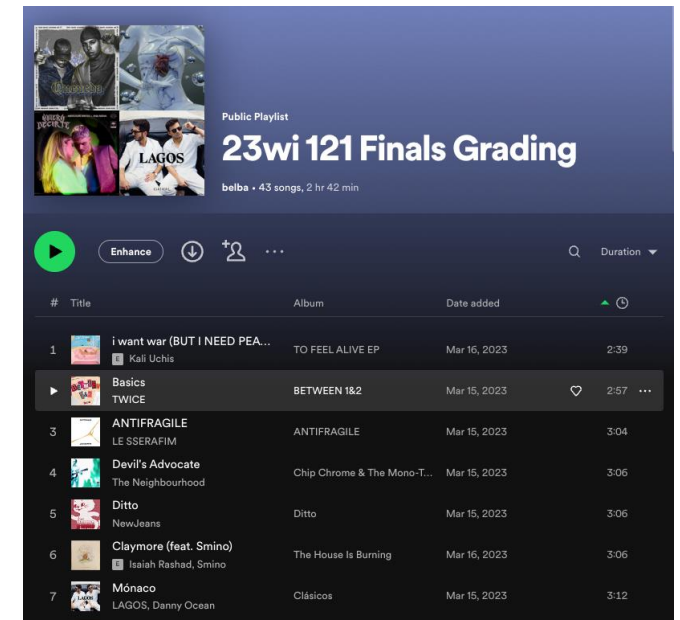
- Students entering CSE 122 are coming from many of different backgrounds
  - UW: CSE 121 or other intro programming course
  - Community College: Intro Programming Course
  - High School Programming Course (e.g., UWHS, AP CS, IB CS, etc.)
  - Self-taught or other previous experience
- Importantly: CSE 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 Creative Project 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? (1/2)

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



122 Data Structures



Wordle





# Why 122? (2/2)

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** 
  - Grading
  - 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.
- Recorded 🕶️

### SECTIONS

(x19)

- Held in person
- More practice, reviews, applications
- TA advice, how to be an effective student
- Preparation for quizzes / exams
- Not Recorded!

## 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 paper

### EXAM

(x1)

- Culminating exam
- **Date/Time TBD**

# Course Website (1/2)

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

**CSE 122**

- Home / Calendar
- Syllabus
- Assignments
- Resubmissions
- Exam
- Staff
- Office Hours
- Grading Rubrics
- COVID-19 Safety
- Resources

---

Course Tools

- EdStem
- Anonymous Feedback
- Code Quality Guide
- Commenting Guide
- Acknowledgements

**Attention!** This website is still **under development**. More information will be added soon and all content is subject to change.

## Introduction to Computer Programming II Autumn 2024

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

- ▶ What is this class? What will I learn?
- ▶ Prior Experience and Expectations

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

**Feedback** Feedback is always welcome! You can contact the the course staff or submit anonymous [feedback](#).

**Registration** Please **do not** email the course staff or instructors regarding registration for the course. The course staff do not have access to add codes. Please email [ugrad-adviser@cs.washington.edu](mailto:ugrad-adviser@cs.washington.edu) for assistance.

### Announcements

### Staff

---

#### Instructor



**Miya Natsuhara** SHE/HER/HERS  
[mnats@cs](mailto:mnats@cs)

**Office Hours**  
Mondays 1:00-2:00pm, Fridays 1:30-2:30pm  
[CSE 460](#) or [Zoom](#)

Hi all, I'm Miya! I've lived in various parts of the Seattle area for all of my life (Renton, Bellevue, Downtown Seattle) and now live in West Seattle. We've been considered a temporary island out here for the past 2.5 years as the West Seattle Bridge has been closed but it's recently opened so we are happy to reunite with the greater Seattle area!

I graduated with a BA in Mathematics along with a BS and MS in Computer Science from the Allen School here at UW. I had no knowledge of programming or Computer Science when I started at UW, and only very reluctantly took the first

1) Instructor  
2) Teaching Assistants  
3) Getting Help

## Get to know the course staff

Contains most course info – check frequently!  
Announcements, Calendar, Lecture Slides, Office Hours schedule,  
Staff Bios, Important Links

# Course Website (2/2)

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

**CSE 122**

- Home / Calendar
- Syllabus
- Assignments
- Resubmissions
- Exam
- Staff
- Office Hours
- Grading Rubrics
- COVID-19 Safety
- Resources

---

Course Tools

- EdStem
- Anonymous Feedback
- Code Quality Guide
- Commenting Guide
- Acknowledgements

**Attention!** This website is still **under development**. More information will be added soon and all content is subject to change.

## Introduction to Computer Programming II Autumn 2024

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

- ▶ What is this class? What will I learn?
- ▶ Prior Experience and Expectations
- Syllabus** If you want to learn more about the course and its policies, please check out our [course syllabus](#).
- Feedback** Feedback is always welcome! You can contact the the course staff or submit anonymous [feedback](#).

**Registration** Please **do not** email the course staff or instructors regarding registration for the course. The course staff do not have access to add codes. Please email [ugrad-adviser@cs.washington.edu](mailto:ugrad-adviser@cs.washington.edu) for assistance.

### Announcements

**Staff**

---

Instructor 1) Instructor

---

**Syllabus**

---

Course Information

Teaching Staff

**Instructor:** Miya Natsuhara & Elba Garza

**Instructor Email:** [cse122-24au-instructors@uw.edu](mailto:cse122-24au-instructors@uw.edu)

**Registration Questions:** CSE Advisers ([ugrad-adviser@cs.washington.edu](mailto:ugrad-adviser@cs.washington.edu))

**Course Staff and Support Hours:** [Course Staff and Office Hours](#)

▼ Who to contact?

*To ensure the security of your personal information, all communication related to this course should be conducted through either the EdStem platform or via*

Contains most course info – check frequently!  
Announcements, Calendar, Lecture Slides, Office Hours schedule,  
Staff Bios, Important Links

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



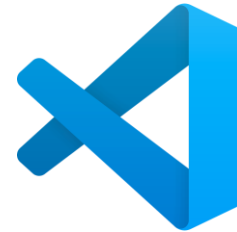
# Other Course Tools



## 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

## VSCode (Optional)

- Develop offline
- Visual debugger



## Canvas


- Lecture recordings



## Sli.do

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

# Lecture Outline

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

# Graded Course Components

- Your grade will consist of the following categories:
- Each mark is graded on the scale:
  - **E**(xcellent)
  - **S**(atisfactory)
  - **N**(ot yet)

Category	#	Marks per	Total Marks
Programming Assignments	4	4 ( <a href="#">Behavior, Concepts, Quality, Testing/Reflection</a> )	16
Creative Projects	4	1	4
Quizzes	3	3 (3 questions)	9
Exam	1	6 (6 questions)	6

# Course Grades

In assigning course grades, we'll use a bucket system:

- Marks earned place in an initial bucket, additional S+ marks improve grade.
- Must meet all requirements of a bucket for initial placement.
- These are minimum GPA guarantees – grade can always be higher than minimum promise. 😊

Minimum Grade	Required S+	Required E
3.5	30	27
3.0	27	22
2.5	24	17
2.0	21	0
1.5	14	0
0.7	8	0

S+ indicates S or E

# Lecture Outline

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



# Resubmissions

*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.
  - An assignment can be resubmitted in the 3 cycles after feedback has been published
  - *Tip: Resubmit as early as possible*

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.
  - Be aware of and avoid use of [Forbidden Features](#) in submitted work
  - Always cite the help you receive on graded work
- **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.

The screenshot shows the Ed platform interface for a lesson titled "Arrays Review". The top navigation bar includes "Lessons", "Slides", "Prev", and "Next". The main content area is titled "Arrays Review" and contains an information icon and a note: "On the left hand side, you'll see there's a lesson titled **ArrayLists [Video Walkthrough]**. The video and the reading both have the same information! You're not required to go through both the video and the reading, as the video just walks through the reading to help contextualize it!". Below this, there is a section titled "Previously in CSE 121, we had learned about **arrays** – a data structure than can hold multiple values of the same type!". This is followed by a paragraph: "As mentioned previously, we like to think of arrays as **cubbies** – or a group of variables that are stored together in one data structure. Remember that arrays have the following (with an accompanying diagram below):". A list of four characteristics of arrays is provided: 1. a *name*, 2. a *specific length* (number of compartments), 3. a *specific type* that each of its compartments can hold, and 4. compartments where each compartment has: an *index* (like `String` indices, starting at index 0) and the ability to hold a piece of data. Below the list, it states: "To initialize an array, you need the following:". A list of three steps is provided: 1. **type[]** – start by listing the type of your array and its elements and make sure to have the opening and 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 follows prescribed naming guidelines. 3. **array construction code** – the remaining code to construct a new array follows the template `new type[int length]`; where the type should match the type listed on the left hand side of the line of code. Below the text, the code `int[] arr = new int[4];` is shown. A diagram illustrates the array: a box labeled "name: arr (int[])" has an arrow pointing to a row of four light blue boxes, each containing the number "0". Below these boxes are the indices "0", "1", "2", and "3" respectively.

# Lecture Outline

- Introductions
- **About this Course**
  - Course Components & Tools
  - Grading
  - 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). 561 of you >>> 35 of us!
  - For serious personal circumstances, you can email Miya and Elba 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 instead.

# Help Us Improve!

- This is a relatively 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 drop low letter grades in 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
  - Grading
  - 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**

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

- 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)

# 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
- Arrays & 2D 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](#)
  - Optional: Introduce yourself in a Video Introduction!
  - Go meet your TA and classmates in Thursday’s quiz section
  - ★ Complete the pre-class material for Friday (see calendar)
  - [Check over syllabus details](#)