

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? Year? Major?
What did you do over summer break?

Instructor: Brett Wortzman

TAs:	Archit	Daniel	Naomi	Sreshta
	Blake P.	David	Neha	Suyash
	Blake R.	Hanna	Nicole	Trisha
	Cady	Isis	Rio	Wesley
	Carson	Jun	Rohan	Yash
	Cole	Landon	Shreyank	
	Connor	Michelle	Somya	

Lecture Outline

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

Hi, I'm Brett! (he/him)

- Associate Teaching Professor
- Frequent intro CS instructor
- Also interested in CS education/pedagogy
- Previously:
 - trained CS teachers (and still!)
 - developed CS curriculum (and still!)
 - taught high school CS
 - worked as a software engineering
- Non-CS hobbies: board games/RPGs, officiating football, announcing robotics competitions



Students

- Currently 418 students registered for the course!
- Strength in numbers
 - With 418 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.

Lecture Outline

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

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](#) very helpful!
- If you want to know if this class is the right fit for you, take the [Allen School Self-Placement Test](#)

Other Similar Courses

Course	Good choice if...
CSE 121	- You've never programmed before AND - You are, or want to be in a major such as CS, CE, ECE, Info, etc. that requires Java programming
CSE 122 (this is us!)	- You've done <i>some</i> programming (roughly one course worth) in any programming language AND - You are, or want to be in a major such as CS, CE, ECE, Info, etc. that requires Java programming
CSE 123	- You've taken CSE 122 (or equivalent) AND - You are, or want to be in a major such as CS, CE, ECE, Info, etc. that requires Java programming
CSE 143	- You took CSE 142 (at UW, through UWHS, or at community college) OR - You took AP CS A (or similar) and feel confident in <i>all</i> the material
CSE 143X	- You have programmed quite a bit before, but <i>not</i> in Java OR - You have lots of extra time to put into learning and tend to pick things up quickly
CSE 160	- You've never programmed before AND - You're interested in data science and analysis OR - You'd rather learn Python than Java* OR - You are, or want to be in a major such as Physics, Bio, Stat, etc. where your primary goal is analyzing data through programming (rather than building software)

Also see: [guided self-placement](#) and [CSE page on introductory courses](#) for more info.

Lecture Outline

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



Digression: My Pandemic Hobby

Amigurumi: Japanese art of creating crocheted or knitted stuffed toys



Digression: My Pandemic Hobby

Amigurumi: Japanese art of creating crocheted or knitted stuffed toys



Digression: My Pandemic Hobby

Amigurumi: Japanese art of creating crocheted or knitted stuffed toys

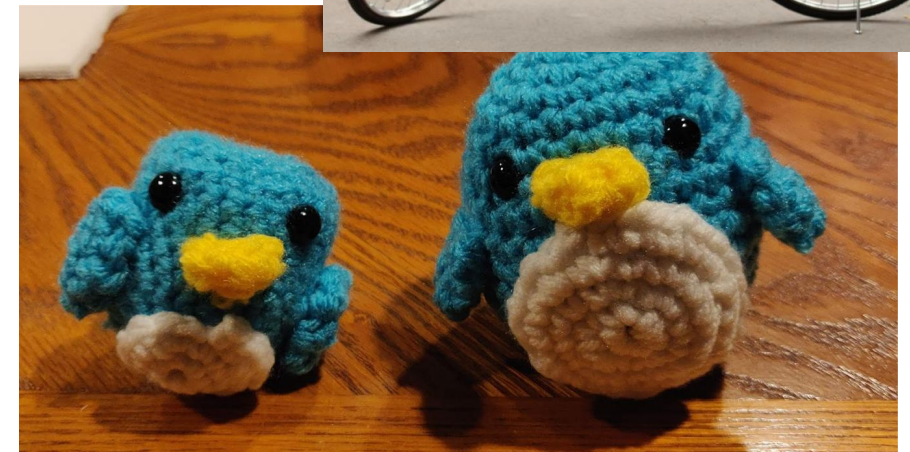


How Learning Works

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**
- Does not work well if you cram everything!



Learning and AI: Our Philosophy

AI amplifies the skills of the AI user

- We cannot teach an introductory programming course without acknowledging the **existence of AI tools**.
- We strongly believe in the value of learning the fundamentals of programming.
- The ability to **understand** and **modify** code is more important than ever.
- AI use is **permitted** on some assignments. AI use is **not required**.

Course Components

PRE-CLASS MATERIALS

x18ish

- Introduce new concepts/ideas
- Readings, sample problems
- Due **before** lecture

LECTURES

x20



- Expand on new concepts
- Apply and practice
- In person, recorded

SECTIONS

x16

- Review concepts
- More apply and practice
- Get TA advice
- In person, **not** recorded

Lessons

Graded Components

Pre-Class Materials

PRE-CLASS MATERIALS

PCMs are a core element of the course

- Prepare for each lecture with **readings & practice problems**
- Should take **~30 minutes per lecture** (why we don't have Monday lectures!)
- Class will start with a brief recap, then pick off where we left off

Pre-Class Materials

PRE-CLASS MATERIALS

PCMs are a core element of the course

- Prepare for each lecture with **readings & practice problems**
- Should take **~30 minutes per lecture** (why we don't have Monday lectures!)
- Class will start with a brief recap, then pick off where we left off

This means...

- We can spend lecture diving deeper, answering questions, and think-pair-share
- You can ask about pre-lecture material in class or section!

Pre-Class Materials

PRE-CLASS MATERIALS

PCMs are a core element of the course

- Prepare for each lecture with **readings & practice problems**
- Should take **~30 minutes per lecture** (why we don't have Monday lectures!)
- Class will start with a brief recap, then pick off where we left off

This means...

- We can spend lecture diving deeper, answering questions, and think-pair-share
- You can ask about pre-lecture material in class or section!

Pre-class materials are **part of your participation grade**, which means...

- It's okay if you find them challenging – that means you're learning!
- But you should do them, and we will assume you've done them

Consistent and Active Participation

ATTENDANCE

Attendance is not graded, but it's strongly encouraged!

- Lectures and sections are not going to be just us talking at you!
- Instead: live in-class coding, debugging, think-pair-share, and problem—solving
- Spending ~1-2 hours each day over Tuesday - Friday is **much more effective** than cramming before the assignment is due!

Consistent and Active Participation

ATTENDANCE

Attendance is not graded, but it's strongly encouraged!

- Lectures and sections are not going to be just us talking at you!
- Instead: live in-class coding, debugging, think-pair-share, and problem—solving
- Spending ~1-2 hours each day over Tuesday - Friday is **much more effective** than cramming before the assignment is due!

Catching up:

- All lectures are recorded on Panopto; slides are on our website.
- Section materials are on Ed, but section will not be recorded

Learning in CSE 122: Live Support Systems

Programming is hard! We **want** to give you collaborative support!

Introductory Programming Lab (TA Office Hours) – starting Week 2

- *many* hours/week (and highly rated in the class!)
- Face-to-face help from TAs on **any** course questions

Instructor Office Hours – starting this week!

- Great for things from lecture, personal questions, or just saying hi

Learning in CSE 122: Async Support Systems

Ed Board

- Best for content and logistics questions – 400+ of you!!
- Encourage public posts, except for things about **your** graded work
- Answer other students' questions – great way to learn!

Email

- Best for personal circumstances and/or private questions
- Unsure? Feel free to email Brett
- May politely ask you to post on Ed instead!
- For emails, **please use your UW email** (protecting student privacy!)

Lecture Outline

- Introductions
- About this Course
 - Making the Most of this Class
 - **Community and Culture** ◀
 - Course Components & Tools
- Intro/Review Java



Practice: Pair

sli.do

#cse122

Think-Pair-Share: Inclusive Environments

CSE 122 will have many think-pair-share activities. Let's practice! Today's think:

What was an experience you had that made you feel welcome or included in a learning environment?

1. **Think** on your own, in silence for about ~ 30 seconds
2. **Pair** with your neighbor about it (and introduce yourself!!)
3. **Share** in sli.do & in class (I'll take a few volunteers from both)



Practice: Pair

sli.do

#cse122

Think-Pair-Share: *Exclusive* Environments

CSE 122 will have many think-pair-share activities. Let's practice! Today's think:

What was an experience you had that made you feel unwelcome or excluded in a learning environment?

1. **Think** on your own, in silence for about ~ 30 seconds
2. **Pair** with your neighbor about it (and introduce yourself!!)
3. **Share** in sli.do & in class (I'll take a few volunteers from both)

The CSE 122 Community

ABOUT US

- Currently ~418 students enrolled!
 - *Only some* are CSE majors
 - Wide range of backgrounds, interests, and goals
 - ***Everyone*** is new to programming

OUR GOALS

- Foster an **inclusive** and **supportive** environment for all students to thrive by:
 - being respectful
 - being kind and understanding
 - being honest
 - being ourselves

The World Around Us

College is challenging and CSE 122 isn't your only class.

Life is unpredictable and things happen.

We can't leave the impacts of the world around us at the classroom door.

OUR POLICIES

Our course policies are **designed for flexibility**:

- Asynchronous help
- Lecture recordings

SUPPORT

We're here to **support you as a student and as a person**.

Please **reach out** if you're struggling or have circumstances that require extra support.

Help Us Improve!

- This is a relatively new course, and we've made lots of changes this quarter!
- 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)

Lecture Outline

- Introductions
- About this Course
 - Making the Most of this Class
 - Community and Culture
 - **Course Components & Tools** ◀
- Intro/Review Java

Course Components

PRE-CLASS MATERIALS

x18ish

- Introduce new concepts/ideas
- Readings, sample problems
- Due **before** lecture

LECTURES

x20



- Expand on new concepts
- Apply and practice
- In person, recorded

SECTIONS

x16

- Review concepts
- More apply and practice
- Get TA advice
- In person, **not** recorded

Lessons

PROGRAMMING PROJECTS

x4

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

COMPUTER QUIZZES

x4

- 45 minutes on computer
- Expand on and connect to projects
- Scheduled outside of class

PAPER QUIZZES

x4

- 45 minutes on paper
- 3 in quiz section, 1 on exam day

PARTICIPATION

x1

- Complete PCMs, participate in section

CODE QUALITY

x3

- Write code well
- Assessed throughout quarter

Graded Components

Course Website

cs.uw.edu/122

- Primary source of course information (**not Canvas**)
- Calendar will contain links to (almost) all resources
- Please **review syllabus ASAP**
- Let's go on a website tour :)

CSE 122

- Home / Calendar
- Syllabus
- Staff
- Office Hours
- 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 2026

Welcome to CSE 122: Introduction to Computer Programming II 

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

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 for assistance.

This Week (at a glance)

Monday (09/28)

- Nothing!

Tuesday (09/29)

- Nothing!

Wednesday (09/30)

- 🧑 Lesson 0: Welcome!
Class sessions @ 12:30pm in ARC 147 (A lecture) and 2:30pm in BAG 131 (B lecture).

Thursday (10/01)

- 🧑 Section 0: Welcome; Java Review

Friday (10/02)

- 🧑 Lesson 1: Java Review & Functional Decomposition
Class sessions @ 12:30pm in ARC 147 (A lecture) and 2:30pm in BAG 131 (B lecture).

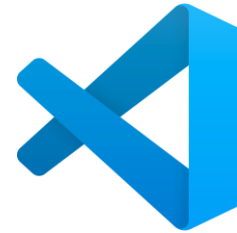
Other Course Tools



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
- Computer quiz scheduling



Sli.do

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

Lecture Outline

- Introductions
- About this Course
 - Making the Most of this Class
 - Community and Culture
 - Course Components & Tools
- **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 HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello world");  
    }  
}
```

- Running on [EdStem](#)

- 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

- [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](#)