LEC 00

CSE 123 Review; Comparable!

Questions during Class?

Raise hand or send here

sli.do #cse123

BEFORE WE START

Talk to your neighbors:

Introduce yourself to your neighbor!

What is your name? Major? What have you been up to the past week?

Instructor: James Wilcox

Lecture Outline

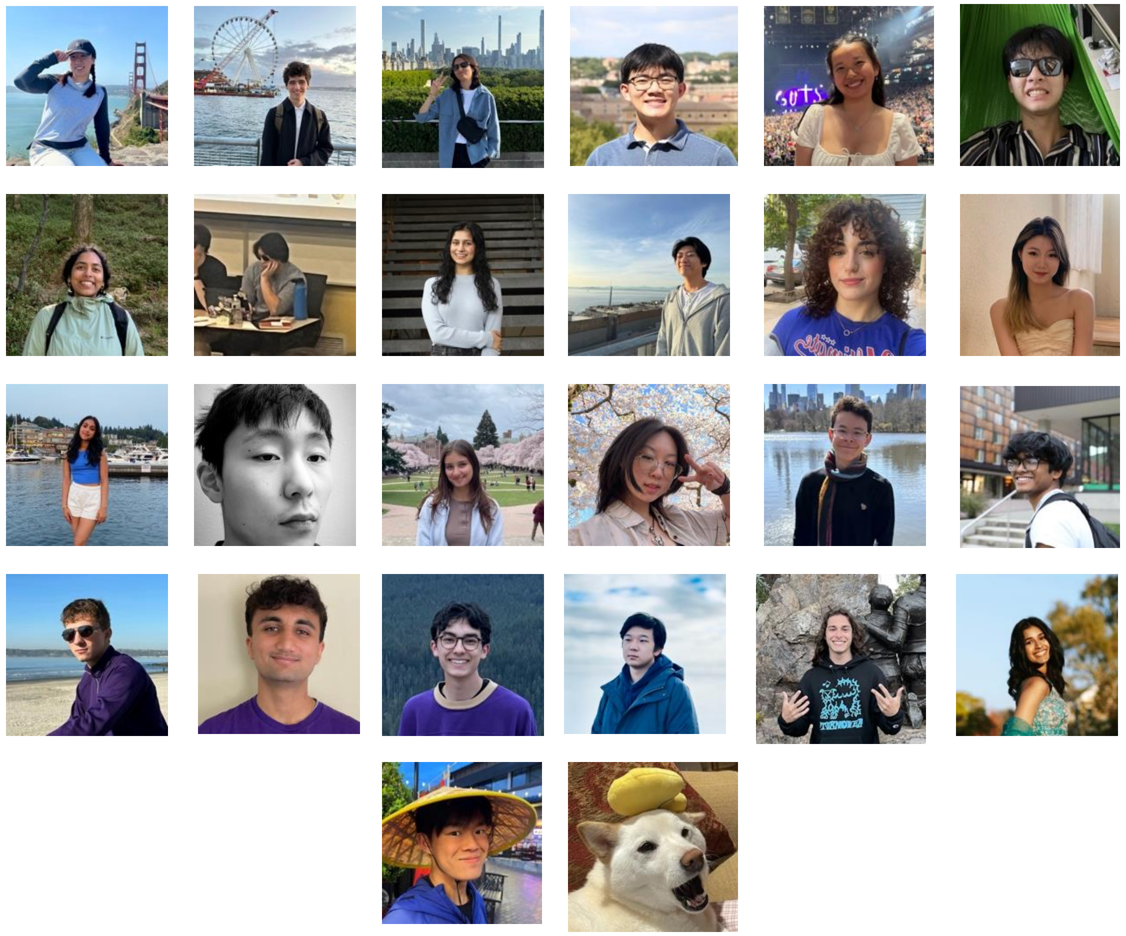
- Introductions
- About this Course
  - Course Components & Tools
  - Making the Most of this Class
- OOP / Junit Review
- Comparable

Course Staff

- Instructor: James Wilcox
  - Feel free to call me "James"

- Teaching Assistants: 31 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 ☺



What is this Class?

CSE 121 – Computer Programming I

- 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

- Functional Decomposition
- File I/O
- Using data structures
  - List, Stacks / Queues, Sets, Maps
- Object Oriented Programming
  - Interfaces

CSE 123 – Computer Programming III

- Advanced Object Oriented Programming
  - Comparable, Inheritance/Polymorphism, Abstract Classes
- Implementing data structures
  - ArrayLists, LinkedLists, Trees
- Recursion
- Critical analysis of design

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 programs
   - Which data structures are "most" appropriate to use

Be a better programmer

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 person

What do you want to get out of this course?

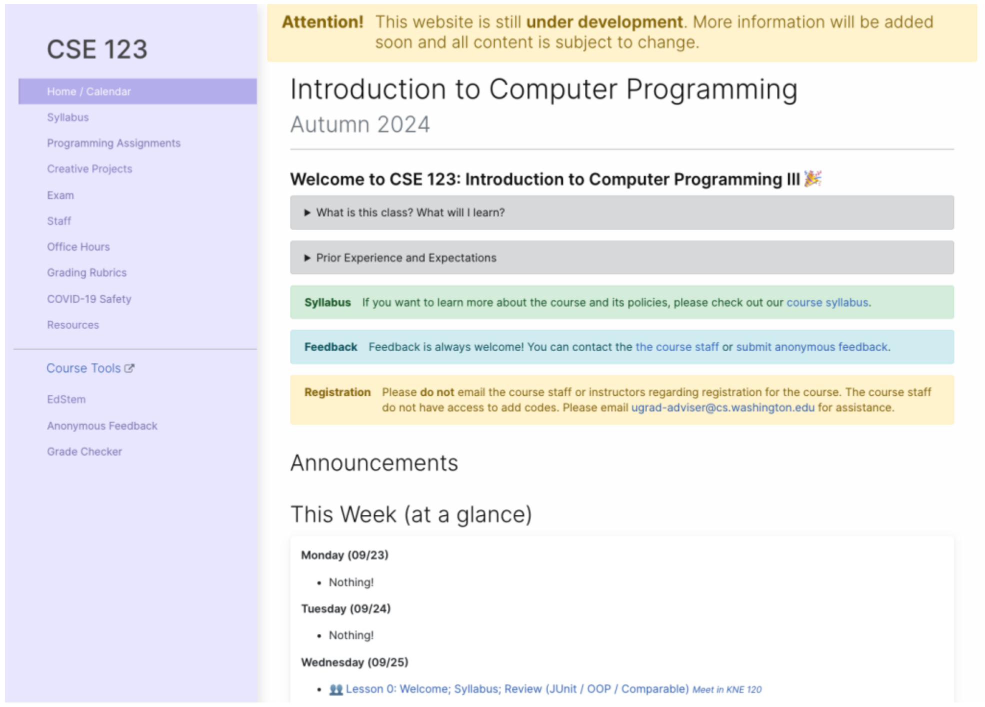
sli.do #cse123

Lecture Outline

- Introductions
- **About this Course**
  - **Course Components & Tools**
  - Making the Most of this Class
- OOP / Junit Review
- Comparable

Course Website

[cs.uw.edu/123](cs.uw.edu/123)



CSE 123

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

Course Tools

- EdStem
- Anonymous Feedback
- Grade Checker

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

Introduction to Computer Programming
Autumn 2024

Welcome to CSE 123: Introduction to Computer Programming III 🎉

▶ 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 for assistance.

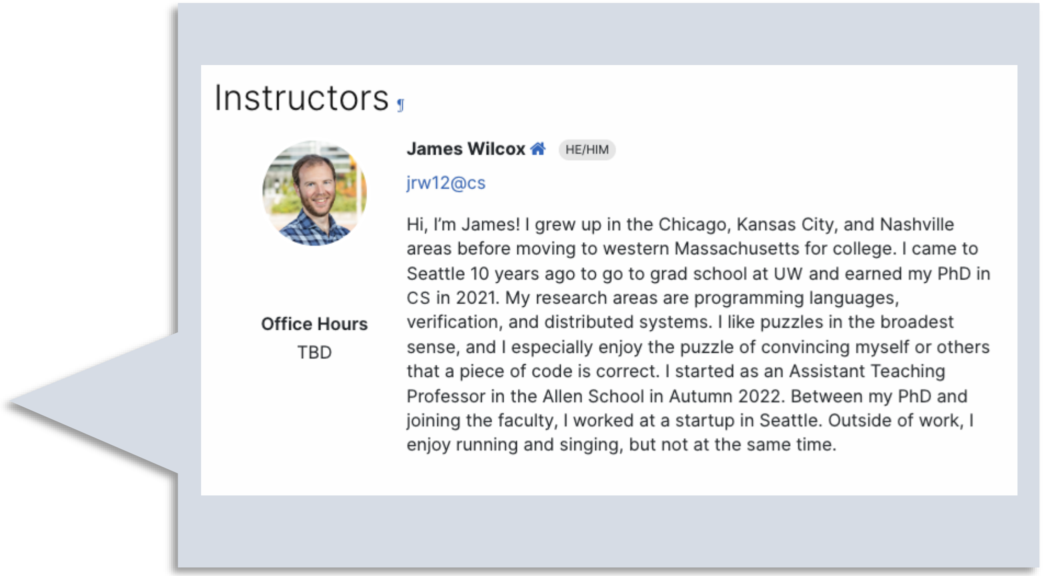
Announcements

This Week (at a glance)

Monday (09/23)
- Nothing!

Tuesday (09/24)
- Nothing!

Wednesday (09/25)
- 👥 Lesson 0: Welcome; Syllabus; Review (JUnit / OOP / Comparable) *Meet in KNE 120*



Instructors

James Wilcox HE/HIM
jrw12@cs

Office Hours
TBD

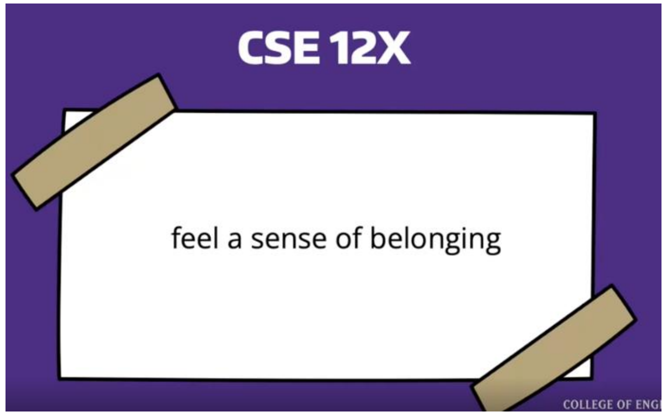
Hi, I'm James! I grew up in the Chicago, Kansas City, and Nashville areas before moving to western Massachusetts for college. I came to Seattle 10 years ago to go to grad school at UW and earned my PhD in CS in 2021. My research areas are programming languages, verification, and distributed systems. I like puzzles in the broadest sense, and I especially enjoy the puzzle of convincing myself or others that a piece of code is correct. I started as an Assistant Teaching Professor in the Allen School in Autumn 2022. Between my PhD and joining the faculty, I worked at a startup in Seattle. Outside of work, I enjoy running and singing, but not at the same time.

Get to know the staff

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

Creating an inclusive environment

Video



- This is a more professional environment than hanging out with friends
- Think about the impact your words can have.
- Collaboration, Support, and Empathy
- Check your own biases and communicate thoughtfully
- Challenge unacceptable behaviors

Other Course Tools

Ed

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

My Digital Hand

My Digital Hand

- Queueing in office hours

VSCode

- 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
  - **Making the Most of this Class**
- OOP / Junit Review
- Comparable

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 outside of class per lesson
  - 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!

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). Many more students than staff!
  - For serious personal circumstances, you can email James 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.

Lecture Outline

- Introductions
- About this Course
  - Course Components & Tools
  - Making the Most of this Class
- **OOP / Junit Review**
- Comparable

Lecture Outline

- Introductions
- About this Course
  - Course Components & Tools
  - Making the Most of this Class
- OOP / JUnit Review
- **Comparable**

Comparable

- `Comparable<E>` is an `interface` that allows implementers to define an ordering between two objects
  - Used by `TreeSet`, `TreeMap`, `Collections.sort`, etc.

- One required method:

```
public int compareTo (E other);
```

- Returned integer falls into 1 of 3 categories
  - `< 0`: `this` is “less than” `other`
  - `= 0`: `this` is “equal to” `other`
  - `> 0`: `this` is “greater than” `other`

$$\underset{\texttt{this}}{\underset{\uparrow}{\mathbf{a}}}\texttt{.compareTo(}\underset{\texttt{other}}{\underset{\uparrow}{\mathbf{b}}}\texttt{);}$$

Subtraction Trick

- `compareTo` implementation when comparing two integers (`a`) ascending:

```
if (this.a < other.a)       -> negative number
else if (this.a > other.a)  -> positive number
else                        -> 0
```

- This is just subtraction!

```
this.a – other.a
```

- What if we wanted to sort descending?

```
other.a – this.a
```

- **Warning**: this only works for integers! Doubles have issues with truncation.