

LEC 01

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?*

Music: [123 24su Lecture Tunes](#) 

Instructor: Joe Spaniac

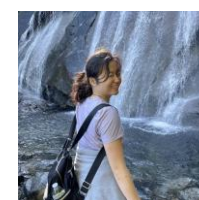
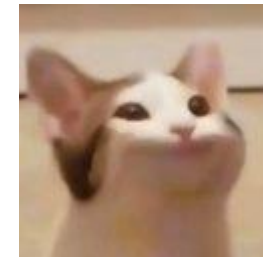
TAs: Andras Daniel Eric Nicole Sahej Trien Zach

Lecture Outline

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

Course Staff

- Instructor: Joe Spaniac
- Teaching Assistants: [7 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 classes to reduce redundancy
 - Which ADT implementations are “most” appropriate to use
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 programmer



Be a better person

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

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

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 Summer 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 (06/17)

Instructors



Joe Spaniac
jspaniac@uw

Office Hours
TBD (CSE 212)

Hey y'all! My name is Joe and I'm currently a 5th years master student in the Allen School. Although I've been a TA for ~15 quarters (all for the intro series here at UW), this is my first time instructing which I'm super excited about! I'm Washington born and raised, originally from the Sammamish / Issaquah area for those familiar, currently calling Ballard home and surviving the commute it entails.

I was privileged enough to have discovered Computer Science my Junior year of high school (IB kids represent), and ever since then I've been hooked. Lately, I've been most interested - unsurprisingly - in CS education and finding ways to distill the crazy buzzwords tossed around (machine learning, AI, computer vision, etc.) into easier to understand concepts for those mostly / entirely unfamiliar with CS (which is a particularly useful skill when trying to explain my HW assignments to my family of non-STEM majors). Outside of school, the last few summers I've interned for AWS and I'm returning as a full-time SWE there next fall. If you want to chat about interviewing for industry or anything else, I'm happy to give my two cents!

Get to know the staff

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

Other Course Tools



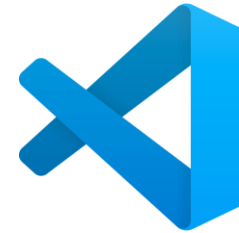
Ed

- 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

- 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 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!




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). 80 of you >>> 8 of us!
 - For serious personal circumstances, you can email Joe 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

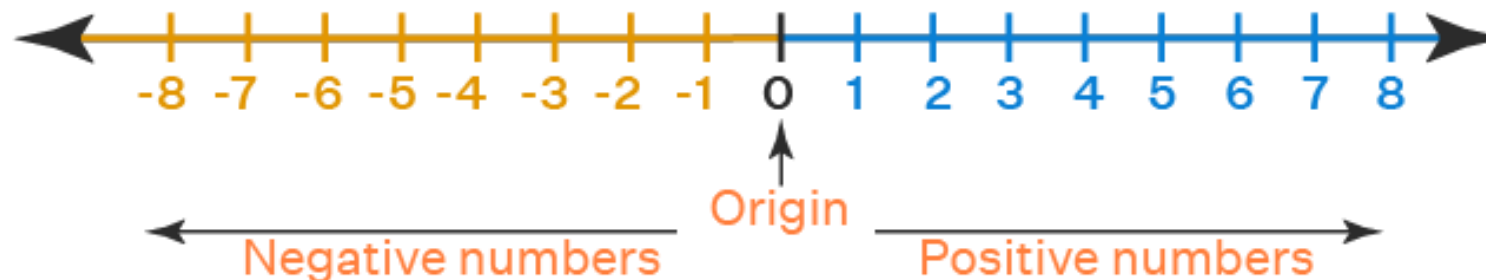
= 0: this is "equal to" other

> 0: this is "greater than" other

`a.compareTo(b)` ;

↑ ↑

this other



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.