Welcome!

Talk to your neighbors:
Introduce yourself to your neighbor!

What is your name? Major? What did you do this summer?

Music: Hunter/Miya’s Playlist

Instructor
Hunter Schafer / Miya Natsuhara

TAs
- Ajay
- Andrew
- Anson
- Anthony
- Audrey
- Chloe
- Colton
- Connor
- Elizabeth
- Evelyn
- Gaurav
- Hilal
- Hitesh
- Jake
- Jin
- Joe
- Joe
- Karen
- Kyler
- Leon
- Melissa
- Noa
- Parker
- Poojitha
- Samuel
- Sara
- Simon
- Sravani
- Tan
- Vivek

Questions during Class?
Raise hand or send here

sli.do  #cse-
Lecture Outline

• Introductions

• About this Course
  - Course Components & Tools
  - Policies
  - Making the Most of this Class

• Intro/Review Java
Course Staff

- Instructor: Hunter Schafer
- Instructor: Miya Natsuhara

- Teaching Assistants: **31 Awesome TAs**
  - Available in section, office hours, discussion board, and 1:1 meetings
  - 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 503 students registered for the course!

• Strength in numbers
  - With 503 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 (Soon™) 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
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 (x19)
- We’re here!
- Introduce concepts, practice ideas, discuss applications.
- Pre-class materials to prepare for class each day. Due before class.

SECTIONS (x18)
- 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 (x4)
- Taken in quiz section
- 30 minutes on computer
- One retake per quiz

EXAM (x1)
- Culminating exam
- Tuesday 12/13 @ 12:30 pm
Course Website

Contains most course info – check frequently!

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

Get to know the staff
Course Website

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
- 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.

• To stay caught-up with the course, each assignment should only be resubmitted at most once over the quarter.
  - If you find an unforeseen circumstance that requires you to use more than one resub for a particular assignment, you need to discuss with your TA a plan to stay caught-up in order before we can accommodate extra resubs.

• 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 (5th 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.
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). 503 of you >>> 33 of us!
  - For serious personal circumstances, you can email Hunter/Miya directly. It never hurts to email us, but if it’s a common logistic question, we will politely tell you to post on the discussion board.

  cse122-22au-instructors@cs.washington.edu
Help Us Improve!

• This is a brand-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

```java
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
- File I/O
- Arrays
In-Class Activities

• **Goal**: Get you actively participating in your learning

• Typical Activity
  - Question is posed
  - **Think** (1 min): Think about the question on your own
  - **Pair** (2 min): Talk with your neighbor to discuss question
    - If you arrive at different conclusions, discuss your logic and figure out why you differ!
    - If you arrived at the same conclusion, discuss why the other answers might be wrong!
  - **Share** (1 min): We discuss the conclusions as a class

• During each of the **Think** and **Pair** stages, you will respond to the question via a sli.do poll
  - Not worth any points, just here to help you learn!
What is the output of this Java program?

```java
public class Demo {
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 9, 14};

        int totalDiff = 0;
        for (int i = 1; i <= nums.length; i++) {
            totalDiff += (nums[i] - nums[i - 1]);
        }
        System.out.println("Total Diff = " + totalDiff);
    }
}
```

A) Total Diff = 12  
B) Total Diff = 11  
C) Total Diff = 7  
D) Error
What is the output of this Java program?

```java
public class Demo {
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 9, 14};

        int totalDiff = 0;
        for (int i = 1; i <= nums.length; i++) {
            totalDiff += (nums[i] - nums[i - 1]);
        }

        System.out.println("Total Diff = " + totalDiff);
    }
}
```

A) Total Diff = 12  
B) Total Diff = 11  
C) Total Diff = 7  
D) Error
“Homework” for Next Time

• First assignment will be released Friday, but there are some things to do in the mean time.

• TODO this week
  - Fill out the introductory survey
  - Post an introduction video on your sections Ed thread! 😊
  - Go meet your TA and classmates in Thursday’s quiz section
  - ⭐ Complete the pre-class material for Friday (see calendar)
  - Check over syllabus details