Welcome to CSE 142!

Ana Jojic
Summer 2022

We’re so excited you’re here!
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model
• Tools and resources
  • Course Website
  • Ed
  • Assessment and grading
  • Collaboration
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

• Tools and resources
  • Course Website
  • Ed
• Assessment and grading
• Collaboration

Lesson 0 - Summer 2022
Hi, I’m Ana! (she/her)

• Instructional Lecturer
• B.S. in Computer Science and minor in Mathematics from UW
• Research in biocomputing and computer vision
• Previously...
  • Frequent TA and Head TA for CSE 142
  • Substituted for CSE 142 lectures

This is my cat

This is not my dog
Meet your TAs
Agenda

• About us

• What is Computer Science?

• About this course
  • Learning objectives
  • Other similar courses
  • Course components

• Our learning model

• Tools and resources
  • Course Website
  • Ed
  • PollEverywhere
  • Discord

• Assessment and grading
• Collaboration

(Fri)
What is Computer Science?

**Algorithm:**
a step-by-step procedure for solving a problem or accomplishing some end especially by a computer

**Programming is like a building block**
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

• Tools and resources
  • Course Website
  • Ed
  • Assessment and grading
  • Collaboration

Lesson 0 - Summer 2022
Learning Objectives

or, “What will I learn in this class?”

• **Functionality/Behavior:** Write functionally correct Java programs that meet a provided specification and/or solve a specified problem

• **Functional Decomposition:** Break down problems into subproblems that are modular and reusable, and define methods to represent those subproblems

• **Control Structures:** Select and apply control structures (e.g. methods, loops, conditionals) to manage the flow of control and information in programs

• **Data Abstraction:** Select and apply basic data abstractions (e.g. variables, parameters, arrays, classes) to manage and manipulate data in programs

• **Code Quality:** Define programs that are well-written, readable, maintainable, and conform to established standards
## Other Similar Courses

<table>
<thead>
<tr>
<th>Course</th>
<th>Good choice if...</th>
</tr>
</thead>
</table>
| CSE 142 | • You’ve never programmed before OR  
          • You’ve done a little programming but feel rusty or not confident AND  
          • You are, or want to be, in major (or looking to take classes) such as CS, CE, EE, Info, etc. that requires Java programming |
| CSE 143 | • You’ve programming in Java before OR  
          • You took AP CS A or IB CS in high school                                                    |
| 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 analyzing data through programming is useful |
Course Components

**Lessons (aka Lectures)**
- MWF, 12:00pm PDT
- Held live in person; recordings released after
- First introductions to course concepts
- Mix of presentation of content and practice activities/problems
- Some required pre-work

**Sections**
- Th, various times
- Led by TAs
- Held live in person; **not** recorded
  - Short videos will be released on occasion when important material is covered
- Additional review, discussion, and practice
- Mostly practice problems

*Attendance is not taken, but you are responsible for all material (including announcements).*
Course Components

Labs (optional)
• T, various times
• Problems released online, support from TAs in person
• CSE 190 sections **X and Y** (*pending open registration*)
• 1 credit course
• Credit/No Credit grading
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

(Fri)

• Tools and resources
  • Course Website
  • Ed
• Assessment and grading
• Collaboration
Learning in CSE 142 (or anywhere)

1. **Exposure**
   - Lessons, videos, textbook
   - Encounter concepts for the first time. See examples and ask questions. Nowhere near mastery!

2. **Guided Practice**
   - Lesson activities, sections, labs
   - Practice with support from course staff. Learn by doing: make mistakes and learn from them. Start to develop mastery.

3. **Independent/Group Practice**
   - Checkpoints, section problems, additional practice
   - Practice on your own or with classmates. Continue to learn by doing. Get close to mastery.

4. **Assessment**
   - Take-home assessments
   - Build on the scaffolding from 1-3. Still learning by doing: you're not done! Demonstrate your mastery (even if it's still developing).
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

(Fri)

• Tools and resources
  • Course Website
  • Ed

(Fri)

• Assessment and grading
• Collaboration
Course Website

cs.uw.edu/142

- Primary source of course information (not Canvas)
- Calendar will contain links to (almost) all resources
Please review the syllabus ASAP.
Ed

- Our online learning platform
- Lessons, sections, labs, assessments all here
- Lecture Megathreads for each day that will be monitored by TAs during lecture
- Intro and walkthrough video forthcoming
You will need to install the JDK and jGRASP
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

• Tools and resources
  • Course Website
  • Ed
• Assessment and grading
• Collaboration
Assessment and Grading

• Our goal in the course is for you to **master the concepts and skills** we teach

• We assess your mastery by asking you to apply the concepts and skills on tasks or problems

• By necessity, we are assessing your *work* as a proxy for your *mastery*

• Your final grade should reflect **the extent to which you have demonstrated mastery of the course objectives**
Assessment

• Your learning in this course will be assessed in four ways:
  • Checkpoints (~weekly)
    • Short quiz on upcoming assessment specification
    • Short problems to help you practice and make sure you’ve got the basics for the week
  • Take-home assessments (~weekly)
    • Large programming assignments to assess your full mastery of that week’s concepts (plus some previous material)
  • Culminating assessments (2/quarter)
    • Series of problems covering all material up to that point
  • Reflections (w/other assignments)
    • Written assignments to help you think critically about your learning and progress
Resubmission

*Learning takes time, and doesn’t always happen on the first try*

- One previous take-home assessment can be **resubmitted** each week
  - Must be accompanied by a write-up describing changes
  - Grade on resubmission will replace original grade

- See the **syllabus** for more details
Grading

*Grades should reflect your mastery of the course objectives*

- Checkpoints, culminating assessments, and reflections are graded **S (Satisfactory) or N (Not Yet)**
  - If you submit on time and meet all requirements, you’ll get an S
- Take-home assessments will be grade **E (Exemplary), S (Satisfactory), N (Not yet), or U (Unassessable)** on four dimensions:
  - Behavior
  - Structure & Design
  - Use of Language Features
  - Code Quality
- Final grades will be assigned based on the **amount of work at each level**

- See the [syllabus](#) for more details
Agenda

• About us
• What is Computer Science?
• About this course
  • Learning objectives
  • Other similar courses
  • Course components
• Our learning model

• Tools and resources
  • Course Website
  • Ed
  • Poll Everywhere
  • Discord

• Assessment and grading
• Collaboration
Collaboration Policy

*Learning is hard, but it’s easier when you learn from each other*

- You are encouraged to form study groups, work together on practice and review, and discuss your ideas and approaches *at a high level*
- If you discuss your ideas with others, you must *cite them*
- All work you submit for grading *must be your own*
- Any work found to not be your own will receive a grade of *U and may not be resubmitted*
  - If it’s not your work, we can’t assess your mastery from it
- See the *syllabus* for more details
Amnesty

Sometimes, we make bad choices that we regret

• “If you submit work that is in violation of the academic conduct policy, you bring the action to Ana’s attention within 72 hours of submission and request amnesty. If you do so, you will receive a grade of U for the initial submission, but you will be allowed to resubmit your work under the normal resubmission process.”

• See the syllabus for more details