Welcome to (the end of) CSE 142!

Ana Jojic

Summer 2022

You Made It!

Thank your TAs!!

Learning Objectives

*or "What ~~will~~ **did** 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

(Partial) Topic List

or another view on "What did I learn in this class?"

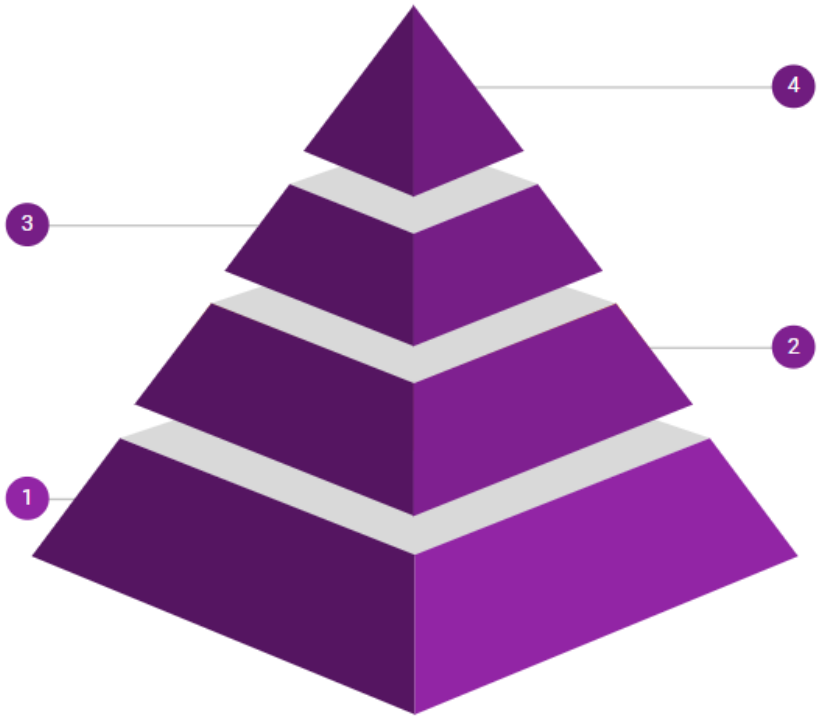
- Methods
- Parameters
- Return Values
- Variables
- Types
- Loops (`for` and `while`)
- Conditionals
- Console (User) I/O
- File I/O
- Arrays
- Classes
- Inheritance
- `ArrayList`
- Many common algorithms!

Underlying Skills

or "What did I learn in this class without realizing it? What skills can I apply anywhere else?"

- **Computational thinking:** breaking problems down into smaller, well-defined steps that can be recombined
  - "Thinking like a computer" (Also called algorithmic thinking)
- **Testing:** determining whether or not a program works as expected
  - Requires really knowing what "as expected" means
- **Debugging:** finding and fixing errors in existing code
  - Often just as hard (or harder!) than writing the code in the first place

Learning in CSE 142 (or anywhere)



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).

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.

Independent/Group Practice
Checkpoints, section problems, additional practice

Practice on your own or with classmates.
Continue to **learn by doing.**
Get close to mastery.

Exposure
Lessons, videos, textbook

Encounter concepts for the first time.
See examples and ask questions.
Nowhere near mastery!

Applications of CS

or "What can I do with what I learned?"

- Detect and prevent toxicity online
- Digitize basketball players
- Help DHH people identify sounds
- Figure out how to best distribute relief funds
- Recognize disinformation online
- Make movies
- Improve digital collaboration
- Fix Olympic badminton
- And so much more!

Future Courses

or "What can I do next?"

Course	Overview
CSE 143 * +	Intermediate programming with data structures (Java) – object-oriented
CSE 122 * + (new)	Intermediate programming with data structures (Java) – client-based
CSE 154 * +	Introduction to web programming (several languages)
CSE 160 * +	Introduction to programming for data analysis (Python)
CSE 163 +	Intermediate programming for data analysis (Python)
CSE 180 * +	Introduction to data science (Python)

* Offered in Fall 2022
+ Offered in Winter 2022 (for CSE 180 under INFO and STAT)
See also: https://www.cs.washington.edu/academics/ugrad/nonmajor-options/intro-courses
For info on new 12x series: https://www.cs.washington.edu/academics/ugrad/nonmajor-options/cse12x

More Java?

Take the adaptive self-placement quiz: https://placement.cs.washington.edu/

With CSE 142 knowledge

Recommended if you started with 142

If you want more practice with writing programs before you start implementing objects and don't mind using 2 quarters to finish the new intro series

CSE 143 (at minimum Fall 2022 – Spring 2023)

- Content covered includes:
  - Object-oriented design, style, and documentation
  - Inheritance vs Interfaces
  - Data-structures use and implementation (lists, stacks, queues, sets, maps, trees, priority queues)
  - Run-time complexity
  - Basic search and recursive algorithms
- Uses same textbook as 142, continues where 142 leaves off (with ArrayList)
- Uses very similar style guidelines to 142

CSE 122 (Fall 2022, and onward)

- Content covered includes:
  - Program design, style, and documentation
  - Interfaces vs Implementation
  - Data-structures use (including lists, stacks, queues, sets, maps)
- Assumes basic programming knowledge
- New course with its own set of guidelines

CSE 123 (Winter 2022, and onward)

- Content covered includes:
  - Object-oriented design, style, and documentation
  - Inheritance vs Interfaces
  - Implementation of data structures (including the above + linked references and trees)
  - Run-time complexity
  - Basic search and recursive algorithms
- New course with its own set of guidelines

With CSE 122 knowledge

Frequently Asked Questions

- How can I get better at programming?
  - Practice!
- How can I learn to X?
  - Search online, read books, look at examples
- What should I work on next?
  - Anything you can think of! (Here are some ideas)
  - Beware: it's hard to tell what's easy and what's hard.
- Should I learn another language? Which one?
  - That depends– what do you want to do?
- What's the best programming language?
  - 😠 (take CSE 341)

Thank you!!!