COS Crash Course: Week 5

126 tips and tricks

name game

Yacoub Kahkajian *he/him*

Graduating in 2026

A.B. Computer Science

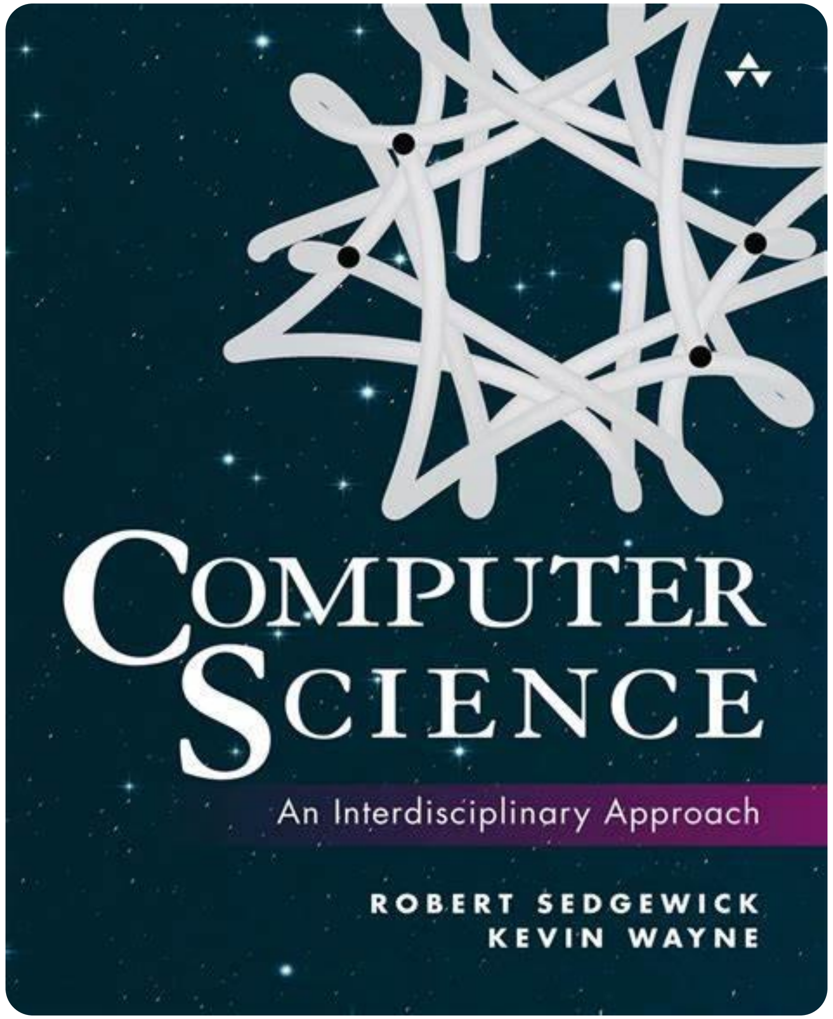
yacoub.xyz

disclaimers

1. I went into 126 with prior programming experience.
2. Things may have (slightly) changed by the time you take the course.
3. Professors may switch between fall and spring semesters.

lecture

during class
assignments
exams



feeling bookish?

caught up in semantics

- Lectures are pre-recorded videos that go over concepts in class. These are optional.
- Class meetings are actual lectures. Weekly overview of topics and hints for assignments.
- Precepts are small-group meetings. Smaller review and group(?) exercises.

class meetings

- Weekly, take place in McCosh.
- Attendance recorded using iClicker.
- Begin with problem to solve, then overview of topic, then examples in Java specific to assignment.
- Slides uploaded onto course website. Notes still helpful for exam review.
- Kahoot-type pop quizzes throughout. Not graded.

Very banking-model-esque!
Friere would be so sad!

precepts

- Small group review sessions
- Occur twice a week
- Work with partner to completed ungraded exercises on Ed
- Typically asked to trace code and recreate algorithms
- Exam reviews when tests approach
- There *may* be a SIFP-specific group you can sign up for this year

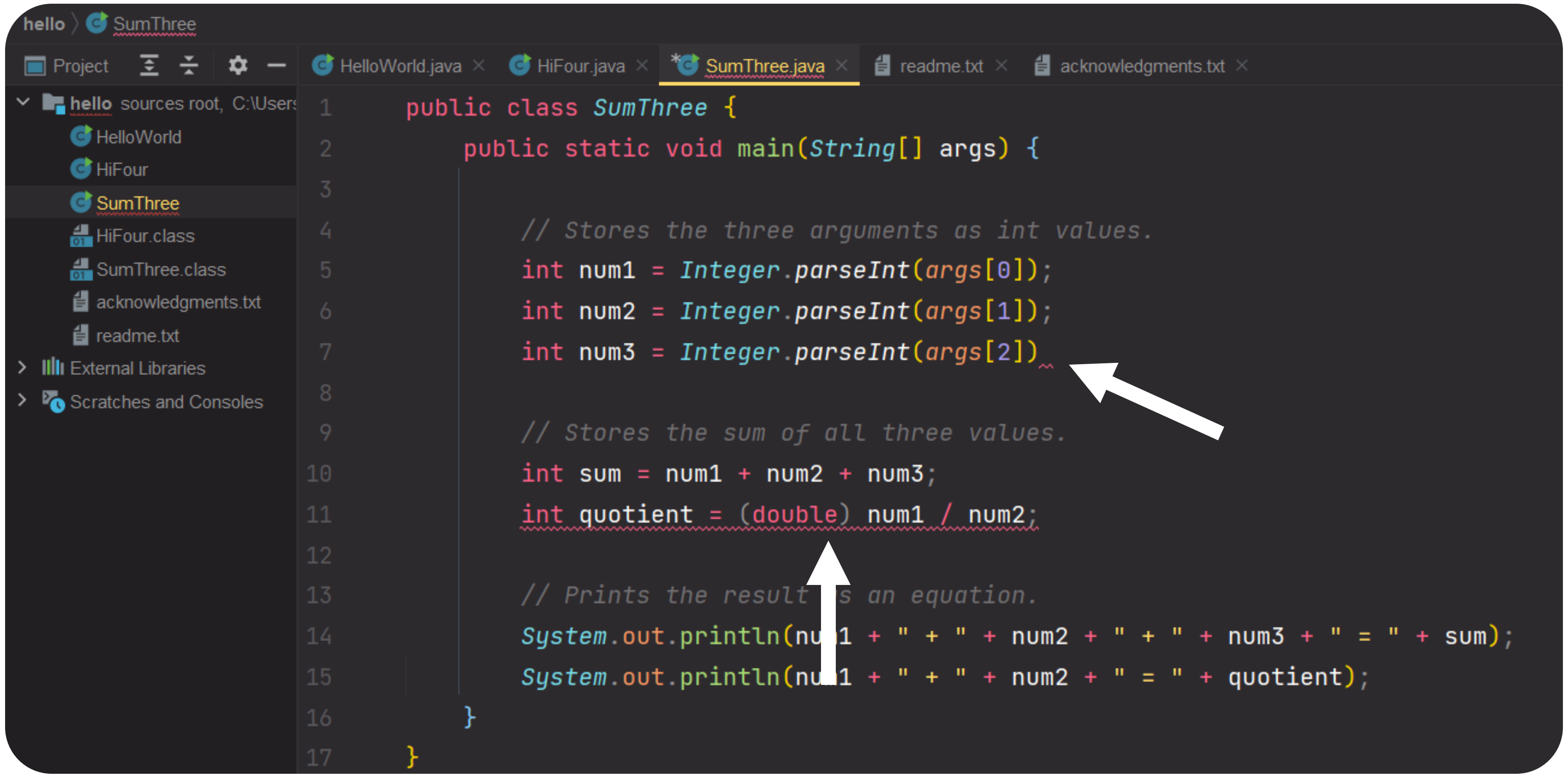
Remaking a binary search has never been so fun!

lecture

precept

assignments

exams



```java
public class SumThree {
    public static void main(String[] args) {

        // Stores the three arguments as int values.
        int num1 = Integer.parseInt(args[0]);
        int num2 = Integer.parseInt(args[1]);
        int num3 = Integer.parseInt(args[2])

        // Stores the sum of all three values.
        int sum = num1 + num2 + num3;
        int quotient = (double) num1 / num2;

        // Prints the result as an equation.
        System.out.println(num1 + " + " + num2 + " + " + num3 + " = " + sum);
        System.out.println(num1 + " + " + num2 + " = " + quotient);
    }
}
```

intellij is neat

what you will/won't do

WILL

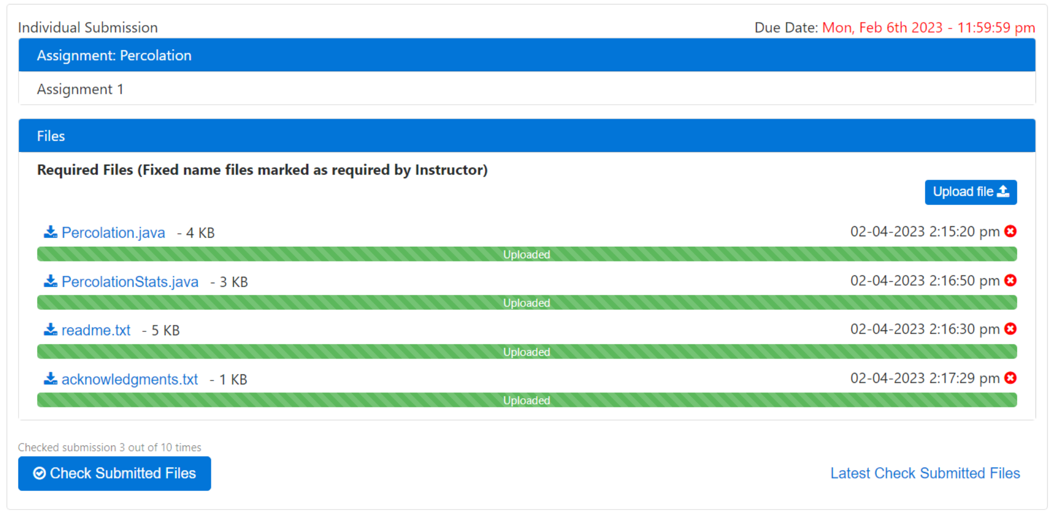
- Follow a program "spec"
- Implement multiple functions
- Use libraries specifically designed for the course
- Write test cases in main
- Answer reflections and follow-up questions in a readme file
- Reference course material (slides, Ed lessons, book or booksite, TAs)

WON'T

- Program one algorithm to solve a single specific problem.
- Be graded on efficiency
- Use non-standard libraries outside of Princeton's
- Email questions (use Ed!)
- Cheat (cue spiel)

submitting code

- Submit code to TigerFile
- No submit button, code "submitted" after readme is uploaded
- TigerFile runs hundreds of test cases on submitted files
- Result returned as a text file with number of cases passed for various tasks



Individual Submission

Due Date: Mon, Feb 6th 2023 - 11:59:59 pm

Assignment: Percolation

Assignment 1

Files

Required Files (Fixed name files marked as required by Instructor)

Percolation.java - 4 KB	02-04-2023 2:15:20 pm
Uploaded	
PercolationStats.java - 3 KB	02-04-2023 2:16:50 pm
Uploaded	
readme.txt - 5 KB	02-04-2023 2:16:30 pm
Uploaded	
acknowledgments.txt - 1 KB	02-04-2023 2:17:29 pm
Uploaded	

Checked submission 3 out of 10 times

Be happy you don't have to mess with Git. Yet.

assignment tips

- While reading the specifications of each assignment, try to find its “Progress Steps” which gives you step-by-step suggestions on how to develop the program. These can be pretty hidden sometimes.
- There’s no limit to how many times you can check your code in 126. Analyzing the specific test cases programs failed is a starting point to find where your code went wrong.

lecture

precept
assignments
exams

more semantics yay

PROGRAMMING EXAM

- Application-based, bring a laptop
- More like a timed mini-assignment than an interview problem
- Same tools as assignments
- Can reference any course material
- Most points given for implementation
- Will likely need to think of an algorithm on the spot for full credit

WRITTEN EXAM

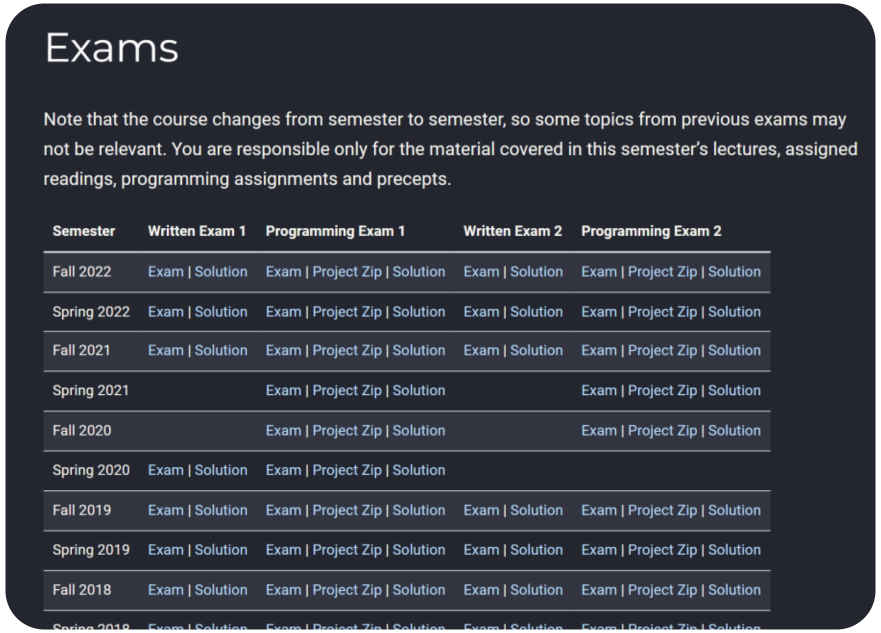
- Theory-based, no technology use
- All multiple choice
- Can reference a one-page handwritten cheat sheet
- Questions about tracing and debugging code
- Questions about Java's quirks

common midterm topics

- Expressions
- Data type properties
- Type conversion
- Tracing loops
- Course-specific libraries
- Previous assignments

studying

- Exams are not curved, your % is your %
- Grinding past exams is the best way to study for upcoming ones
- Get used to the format of written exams
- Practice proctored programming
- Find patterns to solve common problems
- Fill out that cheat sheet!
- Course site has a dedicated exam archive tab

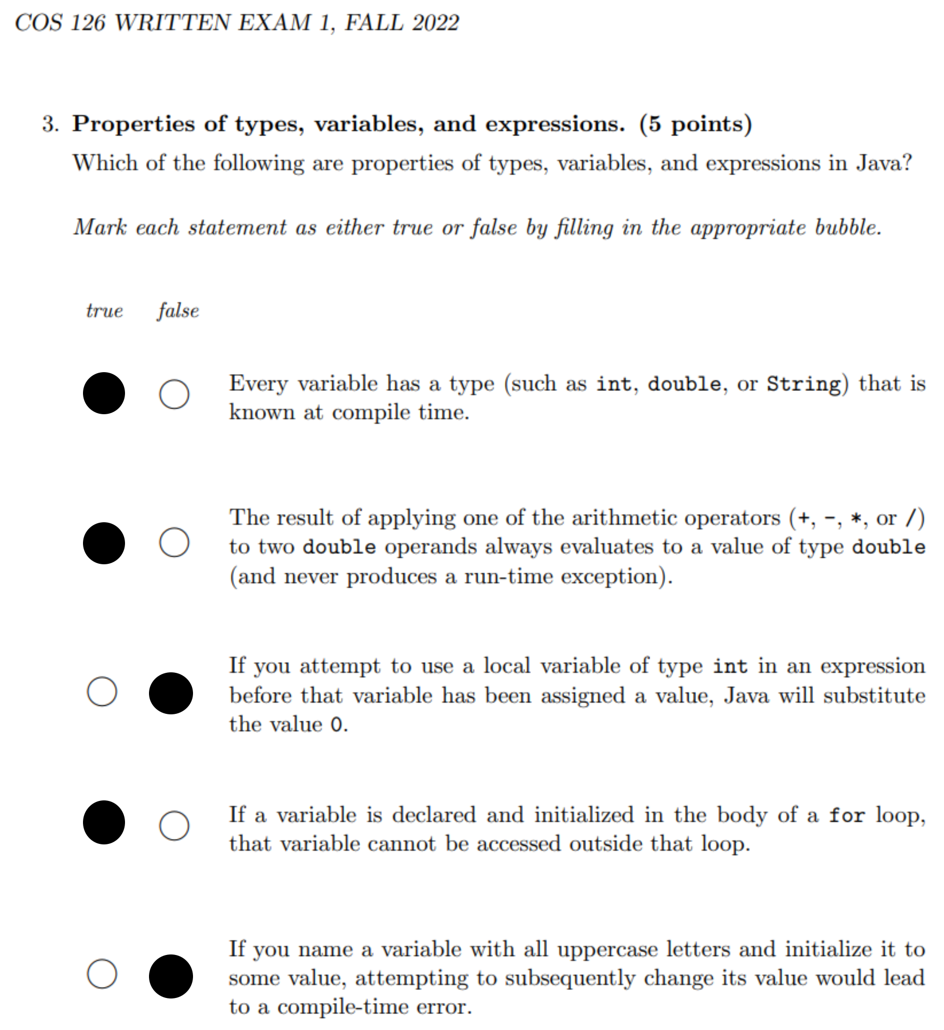


Exams

Note that the course changes from semester to semester, so some topics from previous exams may not be relevant. You are responsible only for the material covered in this semester's lectures, assigned readings, programming assignments and precepts.

Semester	Written Exam 1	Programming Exam 1	Written Exam 2	Programming Exam 2
Fall 2022	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Spring 2022	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Fall 2021	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Spring 2021		Exam | Project Zip | Solution		Exam | Project Zip | Solution
Fall 2020		Exam | Project Zip | Solution		Exam | Project Zip | Solution
Spring 2020	Exam | Solution	Exam | Project Zip | Solution		
Fall 2019	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Spring 2019	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Fall 2018	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution
Spring 2018	Exam | Solution	Exam | Project Zip | Solution	Exam | Solution	Exam | Project Zip | Solution

Princeton's most valuable historical collection.



COS 126 WRITTEN EXAM 1, FALL 2022 3

3. **Properties of types, variables, and expressions. (5 points)**
Which of the following are properties of types, variables, and expressions in Java?

Mark each statement as either true or false by filling in the appropriate bubble.

true	*false*	
●	○	Every variable has a type (such as `int`, `double`, or `String`) that is known at compile time.
●	○	The result of applying one of the arithmetic operators (`+`, `-`, `*`, or `/`) to two `double` operands always evaluates to a value of type `double` (and never produces a run-time exception).
○	●	If you attempt to use a local variable of type `int` in an expression before that variable has been assigned a value, Java will substitute the value `0`.
●	○	If a variable is declared and initialized in the body of a `for` loop, that variable cannot be accessed outside that loop.
○	●	If you name a variable with all uppercase letters and initialize it to some value, attempting to subsequently change its value would lead to a compile-time error.

let's take our own advice.

take a breather.

Get stake in this virtual insanity.