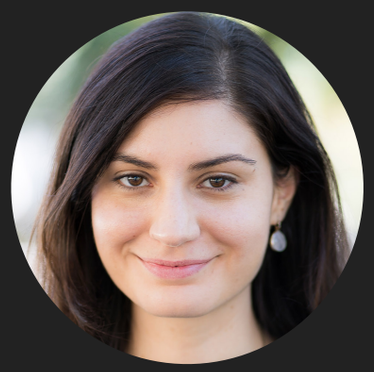
FUNDAMENTALS

CS62

DATA STRUCTURES AND ADVANCED PROGRAMMING

1: Introduction and Java Basics



Alexandra Papoutsaki
she/her/hers

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operators

Some slides adopted from Princeton C0S226 course, Algorithms, 4th Edition, Oracle tutorials

Our team

Neil Chulani
he/him/his

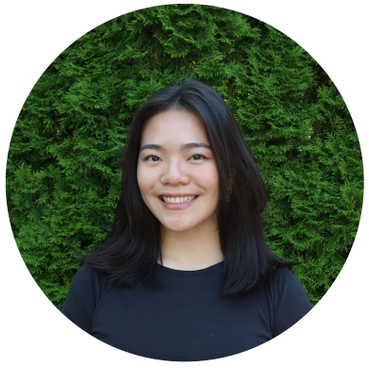
Evelyn Hasama
she/her/hers

Stevie Kim
he/him/his

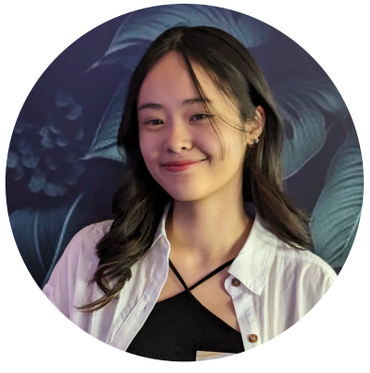
Maxim Koretsky
he/him/his

Camden Le
he/him/his

Gloria Lee
she/her/hers



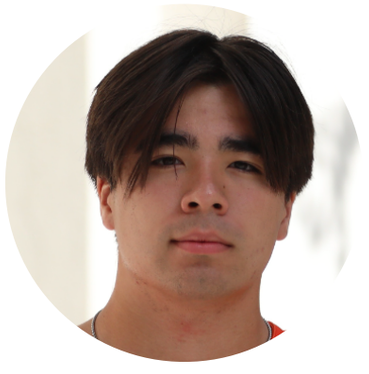
Michele Tang
she/her/hers



Chau Vu
she/her/hers

Emily Wang
she/her/hers

Earn Wonghirundacha
she/her/hers



Alex Wood
he/him/his

Slack Channels

- If registered, already invited to cs62-sp2024 channel in Pomona College Students Slack workspace.
    - You will find the invitation in the Pomona Students workspace http://slack.pomona.edu
- Department-wide slack workspace: https://tinyurl.com/PomonaCSSlack

Who are you?

- Name
- Year
- Programming Experience
  - CS51P
  - CS51A
  - Skipped CS51 because of AP or something else

Sakai Survey Due this Friday at midnight

- “Getting to know you”

What is CS62?

- Beginner to intermediate-level course
- Data structures: Emphasis on how to organize data in a computer based on problem needs
- Advanced Programming: Emphasis on how to write efficient algorithms for modern applications following the Object-Oriented Programming (OOP) paradigm

By the end of this course you will…

- Be familiar with the most commonly used data structures and the time complexity of common operations they support.
- Be able to determine which data structure is appropriate to use based on the time and memory needs of your application.
- Be a more confident programmer, comfortable in reading and writing Java code, and familiar with basic Object Oriented Programming principles.

The advanced programming side of CS62

- In contrast to CS51, labs and assignments will typically be different.
- Labs are shorter and deliverables are due Friday midnight.
- Assignments are week- or two weeks-long, due on Thursday midnight.
- Some assignments will be partner assignments.
- Labs will mostly teach you tools:
  - VS Code, Debugger, Unit testing, git, CLI.
- Assignments will be deliberately vague and will be **using** appropriate data structures to solve interesting problems.
  - Realistically, no one will hire you and give you the steps to solve a problem.
  - But we are here to help you understand how to approach problems!

How can I succeed in CS62?

- Sleep well the night before, eat, come to class, be on time
- Take notes, participate, fill the worksheet, ask questions, don't stay confused
- Review slides and do the assigned reading/problems after each lecture
- Start the assignments early
- Use the tools we learn in the lab (e.g., Debugger)
- Practice writing code on paper
- Learn how to read and write documentation
- Come to office hours/mentor sessions
  - But ask for help *after* you have tried solving a problem by yourself
- Did I say start early?

How can I be a good citizen in CS062?

- Use laptops/tablets/phones/other fancy electronics only for note taking.
- Be mindful when in office hours/mentor sessions of other students waiting for help.
  - Come with specific questions!
  - TAs are students, too. Respect their time outside mentor sessions.
- We encourage collaboration but we want you to submit your own code.
- We monitor assignments for plagiarism. This includes using code from other students, websites, or tools like chatGPT.
  - Academic dishonesty cases are reported to the Dean of Students. Assignments will receive a zero. Exams will receive a zero and half a grade is reduced. Second infraction leads to failure of the course.
  - If unsure about what's allowed, talk to me.

What will my average week look like?

- MW lectures.
- Monday quizzes, starting next week.
- Weekly assignments due on Thursday midnight.
- Friday labs (mandatory) due on Friday midnight.

BUDGET AT LEAST 8 HOURS OUTSIDE THE CLASSROOM

Grading summary

- Weekly Programming Assignments: 30%
  - Three free days - can use on one assignment or across different assignments. Let me know before the deadline if you will take a late day pass.
    - If group assignment, both partners have to use a free day
- Midterm I: 15%
- Midterm II: 15%
- Final Exam: 25%
- Quizzes: 10%
- Labs: 5%
  - No late submissions

More information: http://www.cs.pomona.edu/classes/cs62/

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operators

Java

- One of the most popular general-purpose programming languages.
- Java code is written in .java files. Each Java file has one Java class which matches the name of the file.
  - e.g., Lecture1 . `java` will have a `Lecture1` class where we'll write all of our code.
- In order to run a Java program, we will need a special `main` method.
  - We will ignore both classes and the `main` method for the next two lectures.
- We will use VS Code as an IDE (Integrated Development Environment).
- In contrast to Python, we will use curly braces (`{}`) instead of tabs to create logical blocks of code.
- Single-line comments follow `//` and multi-line are enclosed within `/**/`.

Example Java file `Lecture1.java`


```
public class Lecture1 {
    public static void main(String[] args) {
        //This is a comment

        /*
        * This is a multi-line comment.
        * So much easier than Python
        */
    }
}
```

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operations

Declaring and initializing variables

- Java is statically-typed: all variables must first be declared before use:
  - `dataType variableName = value;`
- For example:
  - `int numberOfCS62Students = 38;`
    - `int` means it can hold integers, that is positive and negative whole numbers.
    - The name of the variable is `numberOfCS62Students`.
    - = assigns the value on the right to the variable on the left.
    - The variable is initialized to 38.
    - I always need to finish a statement in Java using a semi-colon (;).

Assigning new values

- Once a variable is declared, I can reference it elsewhere in the program and assign to it a new value.
  - `variableName = newValue;`
- For example:
  - I could change the number of students to 39, if a new student were to join:
  - `numberOfCS62Students = 39;`
- Note that once a variable has been declared, we do **not** declare again its type. But don't forget the semi-colon.

Naming conventions

- Naming variables is very hard. They should be accurately descriptive and understandable to another reader (and to you, days later).
- It should start with a lowercase letter such as `id`, `name`.
- It should not start with the special characters like &, $,_.
- They should be one word. If the name contains multiple words, start it with the lowercase letter followed by an uppercase letter such as `firstName`, `lastName`.
  - This is known as camel-case.
- Avoid using one-character variables such as `x`, `y`, `z`.

https://www.javatpoint.com/java-naming-conventions

PRACTICE TIME - Worksheet Problems 1a-b.

- Declare a variable that stores the number of CS classes you have taken before CS62 at Pomona and initialize it to the appropriate number.
- Now assume you access this variable at the end of this semester. Assign to it the new value that corresponds to the total number of CS classes you will have taken, including CS62 (and potentially CS101).

ANSWER - Worksheet Problems 1a-b.

- You should end up with something like:
  - `int numberOfCSClasses = 2;`
  - `numberOfCSClasses = 3;`

Primitive data types

- In addition to int, Java supports in total eight primitive data types. A primitive type is predefined by Java and is named by a reserved keyword (that means they have a special meaning. e.g., I can't have a variable named int).
- The eight primitive data types are:
  - `byte`, `short`, `int`, `long`, `float`, `double`, `boolean`, `char`.

Reserved words

Reserved Words				
abstract	default	goto	package	synchronized
assert	do	if	private	this
boolean	double	implements	protected	throw
break	else	import	public	throws
byte	enum	instanceof	return	transient
case	extends	int	short	true
catch	false	interface	static	try
char	final	long	strictfp	void
class	finally	native	super	volatile
const	float	new	switch	while
continue	for	null		

Primitive Data Types

Type	Bits	Default	Example
byte	8	0	`byte yearsOld = 10;`
short	16	0	`short pixels = 2;`
int	32	0	`int luckyNumber = 47;`
long	64	0L	`long bigNumber = 4747L;`
float	32	0.0f	`float smallDex = 47.0f;`
double	64	0.0	`double largeDec = 47.0;`
char	16	\u0000'	`char initial = 'a';`
boolean	1	false	`boolean fun = true;`

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

The most important primitive data types to know

- `int` - for integers.
  - e.g., `int numberOfCS62Students = 40;`
- `double` - for decimal-point numbers.
  - e.g., `double temperatureCelsius = 27.5;`
- `boolean` - for the set of values `{true, false}`.
  - `boolean lovingCS62 = true;`
    - Note that in contrast to Python, `true` and `false` are not capitalized.
- `char` - for alphanumeric characters and symbols.
  - `char firstLetter = 'a';`

Strings

- Character strings are not primitive data types but are supported through the `String` class. Note that `String` is capitalized.
- We enclose strings in double quotes. For example:
  - `String name = “Alexandra";`
- Note that single quotes are reserved for the `char` data type.

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operators

Print statements

- The method `System.out.println()` is used to print an argument that is passed to. For example:
  - `System.out.println("Hello World");`
- `System.out.println(name); //will print Alexandra`
- `System.out.println(numberOfCS62Students); // will print 40`
  - Note that in contrast to Python, you do not need to convert non-string arguments to string, this is done automatically.

String concatenation

- Strings are more commonly concatenated with the + operator, as in `"Hello," + " world" + "!"` which results in `"Hello, world!"`

- The + operator is widely used in print statements, e.g.,

  ```
  System.out.println("My name is " + name + "
  and I will be teaching " +
  numberOfCS62Students + " students this
  semester");
  ```

PRACTICE TIME - Worksheet Problem 1c

- Declare and initialize a variable whose type is a primitive and pass it into a print statement, using string concatenation at least once.

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operators

Operator precedence

Operators	Precedence
postfix	`expr++ expr--`
unary	`++expr --expr +expr -expr !expr`
multiplicative	`* / %`
additive	`+ -`
relational	`< > <= >=`
logical AND	`&&`
logical OR	`||`
ternary	`? :`
assignment	`= += -= *= /= %=`

The Simple Assignment Operator

- One of the most common operators that we've already encountered is the simple assignment operator "="; it assigns the value on its right to the operand on its left:
  - int age = 19;
  - int year = 2024;

Arithmetic Operators

- Java operators support addition, subtraction, multiplication, division, and remainder/modulo.

Operator	Description
+	Additive operator (also used for String concatenation)
–	Subtraction operator
*	Multiplication operator
/	Division operator
%	Remainder operator

PRACTICE TIME - Worksheet Problem 2

Assume you are given the following Java code. What would be printed on your screen?

```
int result = 1 + 2;
System.out.println("1 + 2 = " + result);
int original_result = result;

result = result - 1;
System.out.println(original_result + " - 1 = " + result);
original_result = result;

result = result * 2;
System.out.println(original_result + " * 2 = " + result);
original_result = result;

result = result / 2;
System.out.println(original_result + " / 2 = " + result);
original_result = result;
```

"

ANSWER - Worksheet Problem 2

1 + 2 = 3

3 - 1 = 2

2 * 2 = 4

4 / 2 = 2

2 + 8 = 10

10 % 7 = 3

Other Assignment Operator

- The assignment operators +=, -=, *=, /=, and %= are a compound of arithmetic and assignment operators.
- They operate by adding/subtracting/multiplying/dividing/taking the remainder of the current value of the variable on the left to the value on the right and then assigning the result to the operand on the left. E.g.,
- `num1 += num2;` means `num1 = num1 + num2;`

Unary Operators

- Unary operators require only one operand.

Operator	Description
+	Unary plus operator; indicates positive value (not necessary to have)
–	Unary minus operator; negates an expression
++	Increment operator; increments a value by 1
––	Decrement operator; decrements a value by 1
!	Logical complement operator; inverts the value of a boolean

PRACTICE TIME - Worksheet Problem 3

Assume you are given the following Java code. What would be printed on your screen?

```
 int result = +1;
System.out.println(result);

 result--;
System.out.println(result);

 result++;
System.out.println(result);

 result = -result;
System.out.println(result);

 boolean success = false;
System.out.println(success);

System.out.println(!success);
```

ANSWER - Worksheet Problem 3

1

0

1

-1

false

true

Pre vs post-fix operators

- The increment/decrement operators can be applied before (prefix) or after (postfix) the operand.
- The code `result++;` and `++result;` will both end in `result` being incremented by one. The only difference is that the prefix version (`++result`) evaluates to the incremented value, whereas the postfix version (`result++`) evaluates to the original value.
- If you are just performing a simple increment/decrement, it doesn't really matter which version you choose. But if you use this operator in part of a larger expression, the one that you choose may make a significant difference

Pre vs post-fix operators example

```
int i = 3;
i++;
// prints i (4)
System.out.println(i);
++i;
// prints i (5)
System.out.println(i);
// first increments to 6 then prints it (6)
System.out.println(++i);
// first prints i (6) then increments i to 7
System.out.println(i++);
// prints i (7)
System.out.println(i);
```

Equality and relational operators

- Determine if one operand is greater than, less than, equal to, or not equal to another operand

Operator	Description
==	equal to
!=	not equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

Conditional operators

- The && and || operators perform Conditional-AND and Conditional-OR operations on two boolean expressions

exp1	exp2	exp1 && exp2	exp1 || exp2
`true`	`true`	`true`	`true`
`true`	false	false	`true`
false	`true`	false	`true`
false	false	false	false

PRACTICE TIME - Worksheet Problem 4

Consider the following code snippet:

```
int i = 10;
int n = i++%5;
```

a. What are the values of `i` and `n` after the code is executed?

b. What are the final values of `i` and `n` if instead of using the postfix increment operator (`i++`), you use the prefix version (`++i`)?

ANSWER - Worksheet Problem 4

a. `i` is 11, and `n` is 0

b. `i` is 11, and `n` is 1.

Lecture 1: Introduction and Java Basics

- About this course
- Getting started
- Variables
- Print statements
- Operators

Readings:

- Variables: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/variables.html
- Operators: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/operators.html

Code

- Lecture 1 code

Worksheet

- Lecture 1 worksheet