

# Programming Assignment 0

## Warmup and Review

### Overview

This assignment is intended to be a review and warm up for CSE 123. It will require you to use the skills and concepts that you should be familiar with from your prior programming experience. This is designed to help everyone review and practice the programming skills that will be necessary to succeed in CSE 123. While we don't necessarily expect everyone to find this assignment easy, if you find yourself having major difficulties with any of the content, please contact the course staff to get support!



### Learning Objectives

- By completing this assignment, students will demonstrate their ability to:
- Predict the behavior and results of executing a Java program that includes complex and/or compound data
- Identify errors in a Java program's state or behavior, and implement fixes for identified errors
- Write functionally correct Java programs that meet a provided specification using compound data types
- Write functionally correct Java classes to represent new, compound data types



### Assignment Structure

Unlike most future assignments in CSE 123, this assignment will consist of a series of individual questions and problems. By focusing on a few separate and slightly smaller programming problems, we can help you target your practice on the programming skills that will set you up for success in our course.

Don't worry if you don't find this assignment particularly exciting since we are focusing on review here. We will have many very exciting applications of programming in our future assignments!

To complete this assignment, you should go to each slide and complete the task(s). For quiz slides (indicated by a blue clipboard icon), provide an answer to each question. For coding challenge slides (indicated by a yellow angle brackets icon), upload your code to the workspace. When you have successfully completed each slide, you will see the dot next to the slide title fill

in. The assignment is complete when you have a filled-in dot for every slide. The problems can be worked on in any order.

## Feeling Stuck?

While we expect this assignment to be review, it's still OK if you find this assignment a bit challenging! Remember that learning is a challenging process, and you don't have to do it alone!



- You can visit the [Introductory Programming Lab \(IPL\)](#) to talk with a TA about programming concepts or get help on assignments.
- You can stop by [Brett or David's office hours](#) to discuss course concepts or get help on assignments or discuss the course in general.
- You can post questions on the [discussion board](#) here on Ed! You can make questions public (anyone can see them) or private (only course staff can see them). This is a great way to asynchronously get help on an assignment or ask questions about the course.

It is OK to get stuck and feel challenged by this assignment. However, note that this is intended to be a warm-up for the type of programming we will be doing for the rest of the quarter, and the tasks we will be solving in future weeks will be more complex than these problems and rely on a solid grasp of the skills practiced in this assignment. If you feel like you cannot do this assignment at all, we recommend reaching out to Brett and David ([cse123-instructors@cs.washington.edu](mailto:cse123-instructors@cs.washington.edu)) or the CSE undergrad advisors ([ugrad-adviser@cs.washington.edu](mailto:ugrad-adviser@cs.washington.edu)) to discuss more about academic planning and which programming course might be a good fit for your goals.

## Opening coding portions in VSCode

1. Download the CSE123-P0.zip from the course web page.
2. Unzip the zip file:
  - a. Windows: Right-click "Extract all", confirm the dialog that appears.
  - b. macOS: Double click the .zip file
3. You should now have a directory called "CSE123-p0" with a pdf and a folder called p0-code inside of it.
4. In VSCode:
  - a. File->Open Folder
  - b. Select the p0-code directory that contains the various .java files.

## Submission

When you are ready to submit, go to the " Final Submission " slide, read the statement and check the box, then click "Submit" in the upper-right corner. You may submit as many times as you want until the due date.

You can see your previous submissions by clicking the three dots icon in the upper-right and selecting "Submissions and Grades." By default, we will grade your latest submission from before the deadline. However, if you would like us to grade a different submission, you can select that submission on the left side of the window and click "Set final." Note that we will not grade any submission made after the deadline-- if you mark a submission after the deadline as final, we will grade your most-recent on-time submission instead.

Please make sure you are familiar with the resources and policies outlined in the [syllabus](#) and the [programming assignments](#) page.

For the coding portions, you can drag-and-drop the relevant file into Ed.

For example, after fixing Debugging.java in VSCode, you can go to the "Collections/Reference Semantics -Debugging" slide on Ed, and drag your fixed version of Debugging.java onto the right-hand side of the slide. It will then replace the version on Ed with your new one.

Not sure where your files are? Right-click the filename in the tab at the top of VSCode and select "Show in file explorer" and it will help you out.

# Part 1: Code Comprehension

These questions do not involve writing and running code. Please answer them in Ed.

This slide contains a few problems that ask you to read and interpret Java code. Read and answer each question.

The first 4 questions are based on the following method:

```
public static void arrayMystery(int[] a) {  
    for (int i = 1; i < a.length - 1; i++) {  
        a[i] = a[i - 1] - a[i] + a[i + 1];  
    }  
}
```

For each question, indicate what values would be stored in the array after passing that array as the parameter to the method `arrayMystery`.

Write your response as the array would be printed by `Arrays.toString` (i.e. values separated by commas and surrounded by square brackets, such as `[1, 2, 3]`).

1. `[6, 2, 4]`
2. `[6, 0, -1, 3, 5, 0, -3]`
3. `[7, 7, 3, 8, 2]`
4. `[42, 42]`

Assume there exists a class called `Point` that includes a two-parameter constructor. Consider the following code:

```
Point p1 = new Point(1, 4);  
Point p2 = new Point(3, 5);  
Point p3 = p2;  
Point p4 = new Point(3, 5);  
Point p5 = p3;  
Point p6 = p1;
```

How many `Point` objects are created in the above code?

Consider the same code from the previous question:

```
Point p1 = new Point(1, 4);
Point p2 = new Point(3, 5);
Point p3 = p2;
Point p4 = new Point(3, 5);
Point p5 = p3;
Point p6 = p1;
```

How many references to Point objects are created in the above code?

Consider the following method:

```
public static List<String> mystery(List<String> words, int max) {
    List<String> result = new ArrayList<String>();
    for (String word : words) {
        if (word.length() > max) {
            result.add(word);
        }
    }
    return result;
}
```

Which of the following would be the best description of this method for a method comment?

- Returns a list of words.
- Iterates over a given list of strings using a for loop, checking if the length of each string is bigger than a given maximum, and adds it to a new list if so, then returns that new list.
- Returns a new list containing the strings from the parameter list that are more than max characters long.
- Returns a new list containing the strings from the parameter list that are at least max characters long

## Part 2: Collections/Reference Semantics - Debugging

Work in `Debugging.java`

One of the TAs has been programming in Python for too long, and forgot how to code in Java!  
They wrote a solution to the following problem, but accidentally included some bugs:

Write a method called `deepCopy` that takes as a parameter a map whose keys are strings and whose values are lists of integers and that creates and returns a new map that is a copy of the map parameter. For example, given a variable called `map` that stores the following information:

```
{"cse121"=[42, 17, 42, 42], "cse122"=[10, 12, 14], "cse123"=[100, 99, 98, -97]}
```

The call `deepCopy(map)` should return a new map whose structure and content are identical to `map`. Any later modifications to `map` or the lists in `map` following this call should not be reflected in the copy. The map you construct should store keys in alphabetical order. Your method should not modify the contents of the map passed as a parameter. In constructing collection objects, you are required to use the 0-argument constructors.

There are 5 bugs in the program given. Find and fix them all!

## Part 3: Collections - Inverted Index

Work in `InvertedIndex.java`

Write a method called `createIndex` that creates an inverted index for a list of documents. Your method should take one argument, a list of "documents" where each document is represented as a string. Your method should return a map where the keys are individual words that appear in the parameter list of documents and the values are sets of documents in which those words appear.

For example, suppose the variable `titles` contains the following list:

```
[Raiders of the Lost Ark, The Temple of Doom, The Last Crusade]
```

In this case, the call `createIndex(titles)` would return the following map:

```
{ark=[Raiders of the Lost Ark], crusade=[The Last Crusade], doom=[The Temple of Doom], last=[The Last Crusade], lost=[Raiders of the Lost Ark], of=[The Temple of Doom, Raiders of the Lost Ark], raiders=[Raiders of the Lost Ark], temple=[The Temple of Doom], the=[The Temple of Doom, The Last Crusade, Raiders of the Lost Ark]}
```

The keys of the returned map should be case-insensitive (i.e. treat "The" and "the" as the same word). The keys of the returned map should be in sorted order, while the sets in the values should prefer fast lookup speed.

You may assume that the parameter passed in non-null, that each element of the parameter is a non-empty string, and that words in each document are separated by a single space.

**Important:** When writing your class, be sure to follow all guidelines in the [Code Quality Guide](#) and [Commenting Guide](#). Any additional helper methods created, but not specified in the spec, should be declared ***private***.

**Note:** This is essentially how many search engines work! They build up an index mapping "search terms" (which could be more than single words) to "documents" (which could be more than just strings). See [Wikipedia](#) for more information.

## Part 4: Classes/Interfaces - Media

### Work in Book.java

Write a Java class called `Book` that implements the provided `Media` interface and represents a book. For books, the artists are considered to be the author(s).

Your class should have two constructors:

```
public Book(String title, String author)
    Creates a book with the provided title and single author.
```

```
public Book(String title, List<String> authors)
    Creates a book with the provided title and multiple authors.
```

The title and author(s) should not be able to be modified by a client after creation.

In addition to the methods required by the interface, your `Book` class should include a `toString()` method to produce a readable string representation.

If the book has zero ratings, the string representation should be:

```
<name> by [<authors>]: No ratings yet!
```

If the book has at least one review, the string representation should be:

```
<name> by [<authors>]: <average rating> (<num ratings> ratings)
```

The average rating should be rounded to at most two decimal places *in the string representation only*. (The `getAverageRating` method should return the actual average without rounding.)

**Important:** When writing your class, be sure to follow all guidelines in the [Code Quality Guide](#) and [Commenting Guide](#). Any additional helper methods created, but not specified in the spec, should be declared ***private***.



## Part 5: Reflection

These questions do not involve writing and running code. Please answer them in Ed.

The following questions will ask that you practice **metacognition** to reflect on the topics covered on this assignment and your experience completing it. For each question, focus on your plan and/or process for working through the assignment along with the CS concepts. Think about things like how you organized your working time, what sorts of things tended to go wrong, and how you dealt with those errors or mistakes.

Please answer all questions.

### Question 1

Describe your process for finding and fixing the bugs in the Debugging problem. What skills have you learned or practiced to help you when debugging code? How will you use or adapt this process for future assignments?

### Question 2

Choose either the Inverted Index or Media problem: describe how you would go about testing that the code you wrote for that problem is correct and meets the requirements. What specific test cases would you consider? Why are those cases important?

For example, if we were testing the `ArrayIntList` class from class, important test cases might include checking that an empty list has size zero, checking that an element is in the list after being added, checking that the size of the list increases after an element is added, etc.

### Question 3

What skills did you learn and/or practice with working on this assignment?

### Question 4

What did you struggle with most on this assignment?

### Question 5

What questions do you still have about the concepts and skills you used in this assignment?

#### Question 6

About how long (in hours) did you spend on this assignment? (Feel free to estimate, but try to be close.)

#### Question 7

Was any part of the specification or requirements unclear? If so, which part(s), how was it unclear, and how could it have been made more clear?

#### Question 8

[OPTIONAL] Do you have any other feedback, questions, or comments about this assignment?

(Note that we may not be able to respond to questions here, so please post on the message board if you would like a response!)