LEC 01
CSE 122
Java Review & Functional Decomposition

Questions during Class?
Raise hand or send here
sli.do  #cse-122

BEFORE WE START

Talk to your neighbors:
What is your favorite restaurant around UW?

Music: Hunter/Miya’s Playlist

Instructor: Hunter Schafer / Miya Natsuhara
TAs:
- Ajay
- Andrew
- Anson
- Anthony
- Audrey
- Chloe
- Colton
- Connor
- Elizabeth
- Evelyn
- Gaurav
- Hilal
- Hitesh
- Jake
- Jin
- Joe
- Joe
- Karen
- Kyler
- Leon
- Melissa
- Noa
- Parker
- Poojitha
- Samuel
- Sara
- Simon
- Sravani
- Tan
- Vivek
Lecture Outline

• Review Java

• Functional Decomposition

• Code Quality

• First Assignment
  - Grading
Reminders: Review Java Syntax

Java Tutorial reviews all the relevant programming features you should familiar with (even if you don’t know them in Java).

- Printing and comments
- Variables, types, expressions
- Conditionals (if/else if/ else)
- Loops (for and while)
- Strings
- Methods
- File I/O
- Arrays
In-Class Activities

- **Goal**: Get you actively participating in your learning

- **Typical Activity**
  - Question is posed
  - **Think** (1 min): Think about the question on your own
  - **Pair** (2 min): Talk with your neighbor to discuss question
    - If you arrive at different conclusions, discuss your logic and figure out why you differ!
    - If you arrived at the same conclusion, discuss why the other answers might be wrong!
  - **Share** (1 min): We discuss the conclusions as a class

- During each of the **Think** and **Pair** stages, you will respond to the question via a sli.do poll
  - Not worth any points, just here to help you learn!
What is the output of this Java program?

```java
public class Demo {
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 9, 14};

        int totalDiff = 0;
        for (int i = 1; i <= nums.length; i++) {
            totalDiff += (nums[i] - nums[i - 1]);
        }
        System.out.println("Total Diff = " + totalDiff);
    }
}
```

A) Total Diff = 12  
B) Total Diff = 11  
C) Total Diff = 7  
D) Error  

Total Diff = 11
What is the output of this Java program?

```java
public class Demo {
    public static void main(String[] args) {
        int[] nums = {2, 3, 5, 9, 14};

        int totalDiff = 0;
        for (int i = 1; i <= nums.length; i++) {
            totalDiff += (nums[i] - nums[i - 1]);
        }
        System.out.println("Total Diff = "+ totalDiff);
    }
}
```

A) Total Diff = 12
B) Total Diff = 11
C) Total Diff = 7
D) Error
Case Study: Ballot Counting

Want to write a program that prompts the user for a file containing electoral votes for candidates and report information about who won the election.

Review skills

- User input
- File I/O
- Cumulative sum
- Working with doubles

Kiwi 4 2 4
David 3 1 9
Lucy 4 4 4
Maine 4 1 1
Rebecca 1 3 1
Lecture Outline

• Review Java

• Functional Decomposition

• Code Quality

• First Assignment
  - Grading
## Functional Decomposition

Functional decomposition is the process of breaking down a complex problem or system into parts that are easier to conceive, understand, program, and maintain.

### “Bake the cookies”

<table>
<thead>
<tr>
<th>Wet</th>
<th>Dry</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Mix butter and sugar</td>
<td>• Mix in flour, baking soda, and chocolate chips</td>
</tr>
<tr>
<td>• Beat in eggs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Place</th>
<th>Bake</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Make cookie-sized balls of dough</td>
<td>• Bake at 350 degrees Fahrenheit for 10 minutes</td>
</tr>
<tr>
<td></td>
<td>• Let cool after</td>
</tr>
</tbody>
</table>
Functional Decomposition

In our code, functional decomposition often means breaking a task into smaller methods (also called functions).

Example: Ballot counting
- Getting file from user
- Tallying votes
- Reporting results
- Rounding output
Avoid Trivial Methods

Introduce methods to decompose a complex problem, not just for the sake of adding a method.

Bad example:

```java
public static void printMessage(String message) {
    System.out.println(message);
}
```

Good Example:

```java
public static double round(double num) {
    return ((int) Math.round(num * 10)) / 10.0;
}
```

Rule of thumb: A method should do at least two steps
   - Ask yourself: Does adding this method make my code easier to understand?
Lecture Outline

• Review Java

• Functional Decomposition

• Code Quality

• First Assignment
  - Grading
Code Quality

“Programs are meant to be read by humans and only incidentally for computers to execute.” – Abelson & Sussman, SICP

Code is about communication. Writing code with good code quality is important to communicate effectively.

Different organizations have different standards for code quality.
- Doesn’t mean any one standard is wrong! (e.g., APA, MLA, Chicago, IEEE, ...)
- Consistency is very helpful within a group project
- See our Code Quality Guide for the standards we will all use in CSE 122
CSE 122 Code Quality

Examples relevant for this week

• Naming conventions
• Descriptive variable names
• Indentation
• Long lines
• Spacing
• Good method decomposition
• Writing documentation
What does this code do? How could you improve the quality of this code? (No Slido poll)

```java
public static int l(String a, char b) {
    int j = -1;
    for (int a1 = 0; a1 < a.length(); a1++) {
        if (a.charAt(a1) == b) {
            j = a1;
        }
    }
    if (j == -1) return -1;
    else return j;
}
```
What does this code do? How could you improve the quality of this code? (No Slido poll)

```java
public static int l(String a, char b) {
    int j = -1;
    for (int a1 = 0; a1 < a.length(); a1++) {
        if (a.charAt(a1) == b) {
            j = a1;
        }
    }
    if (j == -1) {
        return -1;
    } else {
        return j;
    }
}
```
Lecture Outline

• Review Java

• Functional Decomposition

• Code Quality

• First Assignment
  - Grading
# Graded Course Components

- Your grade will consist of the following categories:
- Each mark is graded on the scale:
  - E(xcellent)
  - S(atisfactory)
  - N(ot yet)

<table>
<thead>
<tr>
<th>Category</th>
<th>#</th>
<th>Marks per</th>
<th>Total Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programming Assignments</td>
<td>4</td>
<td>4 (Behavior, Concepts, Quality, Testing/Reflection)</td>
<td>16</td>
</tr>
<tr>
<td>Creative Projects</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Quizzes</td>
<td>4</td>
<td>3 (3 questions)</td>
<td>12</td>
</tr>
<tr>
<td>Exam</td>
<td>1</td>
<td>6 (6 questions)</td>
<td>6</td>
</tr>
</tbody>
</table>
Course Grades

Instead of curving the class, we’ll use a bucket system:
- Marks earned place in an initial bucket, additional S+ marks improve grade.
- Must meet all requirements of a bucket for initial placement.
- These are minimum GPA guarantees – grade can always be higher than min promise.

<table>
<thead>
<tr>
<th>Minimum Grade</th>
<th>Creative Projects</th>
<th>Programming Assignments</th>
<th>Quiz/Exam Problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Marks</td>
<td>Creative Projects</td>
<td>Programming Assignments</td>
<td>Quiz/Exam Problems</td>
</tr>
<tr>
<td>3.5</td>
<td>All (4) S+; 3 E</td>
<td>All (16) S+; 12 E; 3 Es per dim.</td>
<td>16 S+; 14 E</td>
</tr>
<tr>
<td>3.0</td>
<td>All (4) S+; 2 E</td>
<td>14 S+; 8 E; 2 E per dim.</td>
<td>12 S+; 9 E</td>
</tr>
<tr>
<td>2.5</td>
<td>3 S+; 1 E</td>
<td>12 S+; 4 E; 1 E per dim.</td>
<td>10 S+; 4 E</td>
</tr>
<tr>
<td>2.0</td>
<td>2 S+</td>
<td>10 S+</td>
<td>9 S+</td>
</tr>
<tr>
<td>1.5</td>
<td>1 S+</td>
<td>8 S+</td>
<td>7 S+</td>
</tr>
<tr>
<td>0.7</td>
<td>1 S+</td>
<td>4 S+</td>
<td>4 S+</td>
</tr>
</tbody>
</table>
Programming Assignment 0 – Warm Up

• Released today, due next Thursday (10/6) at 11:59 pm on Ed
  - Can submit as many times as you want before initial submission date with Mark button
  - Build good habits: Don’t “shotgun debug”
  - While you do have a resubmission for this assignment, important to meet due date to get as much feedback as possible.

• Focused on reviewing Java concepts and Functional Decomposition
  - Different structure than most assignments with *multiple smaller problems*
  - Green checkmark on slide means that problem is done. Green checkmark on whole lesson means assignment is fully done.

• See [Grading Rubric](#) for how each dimension is assessed.

• IPL opens Monday!
Getting Help

• Discussion Board
  - Feel free to make a public or private post on Ed
  - We encourage you to answer other peoples’ questions! A great way to learn

• Introductory Programming Lab (Office Hours)
  - TAs can help you face to face in office hours, and look at your code
  - You can go to the IPL with any course questions, not just assignments

• Section
  - Work through related problems, get to know your TA who is here to support you

• Your Peers
  - We encourage you to form study groups! Discord or Ed are great places to do that

• Email
  - We prefer that all content and logistic questions go on the Ed discussion board (even if you make them private). 503 of you >>> 33 of us!
  - For serious personal circumstances, you can email Hunter/Miya directly. It never hurts to email us, but if it’s a common logistic question, we will politely tell you to post on the discussion board.

cse122-22au-instructors@cs.washington.edu
Other tasks for next time

TODO this/next week

- Fill out the introductory survey
- Post an introduction video on your sections Ed thread! 😊
- ⭐ Complete the pre-class material for Wednesday (see calendar)
- Start P0
- Attend quiz section Tuesday!