

CP213

Review Session 1

INTRO TO JAVA AND OOP,
CONSOLE I/O,
FLOW OF CONTROL

Please take this moment to sign in...



UPCOMING EVENTS...

Welcome to LCS Event:

Thursday, September 21st @ 7:00 pm IN N1001

Next Review Session:

Tuesday, October 3 (midterm prep)

HawkHacks 2024 Hackathon

The hackathon is hiring execs!

Follow @wluhawkhacks to learn more, and
scan QR code below to apply!



Deadline to apply is September 24 @ 11:59 pm!



Today's Agenda



REVIEW SESSION 1



1. INTRO TO JAVA AND OOP

Introducing y'all to the language and the paradigm

2. CONSOLE I/O

How to interact with the computer

3. FLOW OF CONTROL

How the program flows between the code

Useful Links and Follow Along Resources

MATH AND STATS LEARNING SUPPORT

This office hosts drop in and directed homework sessions to give you any extra help you need with course content in most lower level CS courses. You can also get their course widget in MyLS.

CODING SANDBOXES

A really good Java code sandbox is Replit.com. Follow along or write code on your own time without setting up files by using one of these.

A really good **Python, Java, HTML**, sandbox (and almost every other language) is Replit.com

<https://replit.com/>

Another good **HTML/CSS/JS** sandbox:

<https://codepen.io/>

Mental Health Resources

We know university is very challenging and difficult. Please don't hesitate to reach out to people if you need help. Listed below are a few useful resources you can access, for additional resources please view the LCS linktree.

SAFEHAWK APP

Connects you with telephone helplines and other mental health resources.

STUDENT WELLNESS CENTRE

Provides comprehensive physical, emotional and mental health services for Waterloo and Brantford Campus students.

DELTON GLEBE COUNSELLING CENTRE:

A holistic counselling facility.

Intro to Java

FAST FACTS:

- CREATED BY SUN MICROSYSTEMS TEAM LED BY JAMES GOSLING IN 1991 (NOW OWNED BY ORACLE)
- JAVA IS AN OBJECT-ORIENTED PROGRAMMING (OOP) LANGUAGE
- JAVA IS A HIGH LEVEL LANGUAGE
- ARITHMETIC OPERATION SUCH AS: ADDITION (+), SUBTRACTION (-), ETC. ALL SUPPORTED IN JAVA
- PRIMITIVE VARIABLE TYPES SUPPORTED IN JAVA:

TYPE NAME	KIND OF VALUE	MEMORY USED	SIZE RANGE
boolean	true or false	1 byte	not applicable
char	single character (Unicode)	2 bytes	all Unicode characters
byte	integer	1 byte	-128 to 127
short	integer	2 bytes	-32768 to 32767
int	integer	4 bytes	-2147483648 to 2147483647
long	integer	8 bytes	-9223372036854775808 to 9223372036854775807
float	floating-point number	4 bytes	$-3.40282347 \times 10^{+38}$ to $-1.40239846 \times 10^{-45}$
double	floating-point number	8 bytes	$\pm 1.76769313486231570 \times 10^{+308}$ to $\pm 4.94065645841246544 \times 10^{-324}$

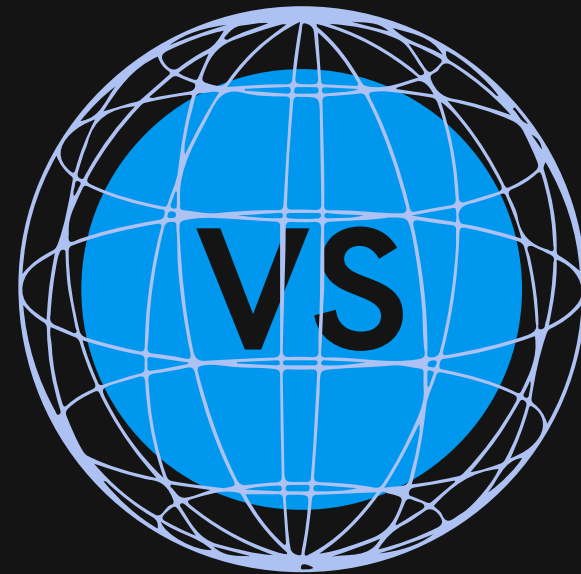
DEFINITIONS:

- HIGH-LEVEL LANGUAGE: READABLE AND WRITABLE BY PEOPLE, REQUIRES TRANSLATION FOR COMPUTER EXECUTION
- MACHINE LANGUAGE: COMPUTER-UNDERSTANDABLE LANGUAGE
- LOW-LEVEL LANGUAGE: MACHINE LANGUAGE OR SIMILAR
- COMPILER: TRANSLATES HIGH-LEVEL LANGUAGE TO LOW-LEVEL LANGUAGE (COMPILING)

What is Object-Orientated Programming (OOP) ?

MODULAR PROGRAMMING

- Code is organized into separate modules or functions, each responsible for a specific task or functionality.
- These modules can be grouped together logically, and the program flow typically involves calling these modules sequentially.



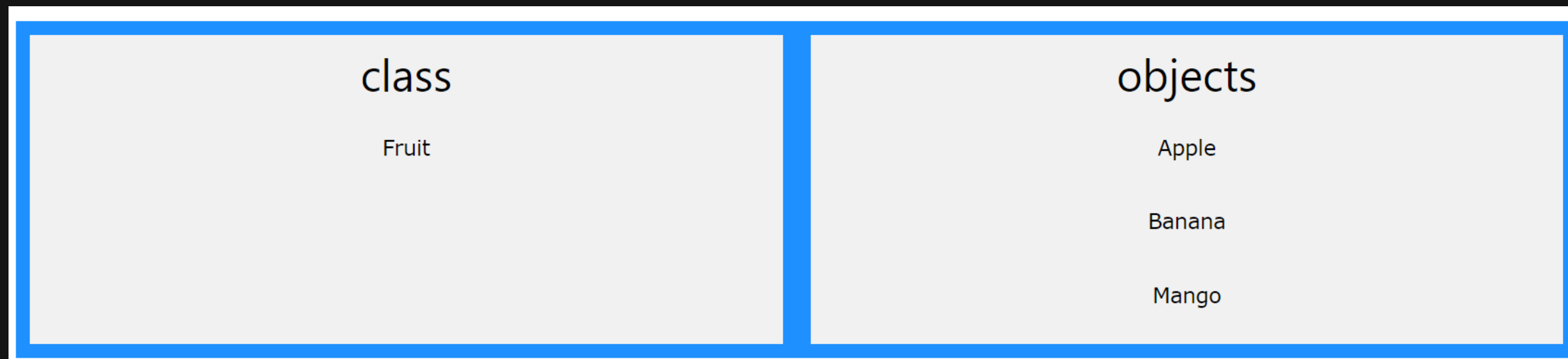
OBJECT ORIENTATED PROGRAMMING

- OOP organizes code around objects, which are instances of classes. Classes define both data (attributes) and behavior (methods).
- Objects encapsulate data and behavior related to a particular concept or entity. The program's structure revolves around defining and interacting with these objects.

In summary...

Modular programming is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.

Ex:



Console I/O: Input

```
package cp213;

import java.util.Scanner;

public class ReviewSession {
    public static void main(String[] args) {
        // Create a Scanner object to read from the standard input (keyboard)
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter your name: ");
        String name = scanner.nextLine(); // Read a line of text

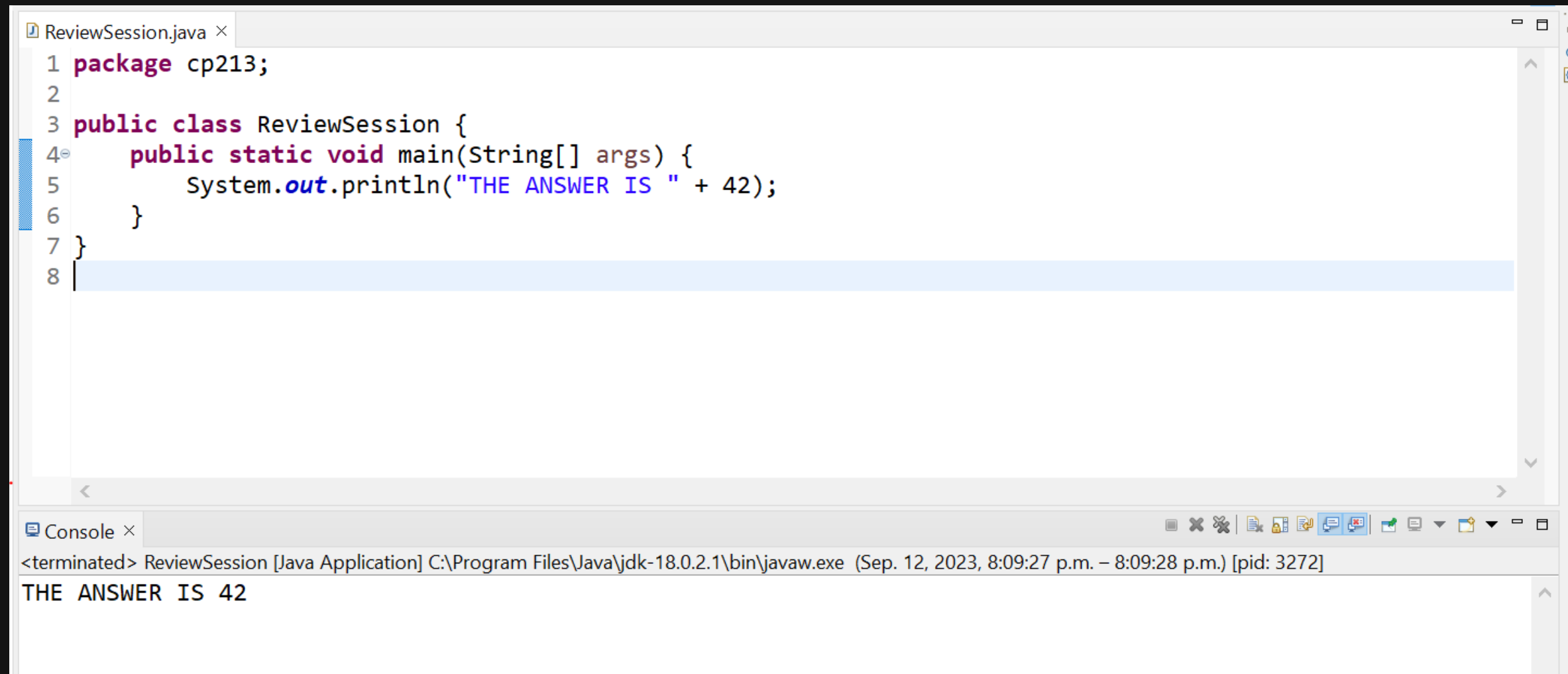
        System.out.print("Enter your age: ");
        int age = scanner.nextInt(); // Read an integer

        System.out.println("Hello, " + name + "! You are " + age + " years old.");

        // Close the scanner when you're done
        scanner.close();
    }
}
```

Console I/O: Output

- **SYSTEM.OUT.PRINTLN FOR CONSOLE OUTPUT**
 - **SYSTEM.OUT** IS AN OBJECT THAT IS PART OF THE JAVA LANGUAGE
 - **PRINTLN** IS A METHOD INVOKED BY THE **SYSTEM.OUT** OBJECT THAT CAN BE USED FOR CONSOLE OUTPUT
 - THE DATA TO BE OUTPUT IS GIVEN AS AN ARGUMENT IN PARENTHESES
 - A PLUS SIGN IS USED TO CONNECT MORE THAN ONE ITEM
 - EVERY INVOCATION OF **PRINTLN** ENDS A LINE OF OUTPUT **SYSTEM.OUT.PRINTLN("THE ANSWER IS " + 42);**
- **OUTPUT:**



The screenshot shows an IDE window with a tab titled 'ReviewSession.java'. The code in the editor is as follows:

```
1 package cp213;  
2  
3 public class ReviewSession {  
4     public static void main(String[] args) {  
5         System.out.println("THE ANSWER IS " + 42);  
6     }  
7 }  
8
```

Below the code editor is a 'Console' window. It displays the output of the program: 'THE ANSWER IS 42'. The console title bar indicates the application is 'ReviewSession [Java Application]' and provides the path to the Java executable and the current time and PID.

Flow of Control

- JAVA HAS BRANCHING MECHANISMS: IF-ELSE, IF, AND SWITCH STATEMENTS
- JAVA HAS THREE TYPES OF LOOP STATEMENTS: WHILE, DO-WHILE, AND FOR LOOPS
- BRANCHING AND LOOPING IN JAVA RELY ON BOOLEAN EXPRESSIONS.
- A BOOLEAN EXPRESSION EVALUATES TO EITHER **TRUE** OR **FALSE**.
- AN IF-ELSE STATEMENT SELECTS BETWEEN TWO OPTIONS BASED ON A BOOLEAN EXPRESSION.
- IF THE BOOLEAN EXPRESSION IS TRUE, IT EXECUTES THE "YES_STATEMENT," OTHERWISE, IT EXECUTES THE "NO_STATEMENT."

```
if (Boolean_Expression)
    Yes_Statement
else
    No_Statement
```

A NESTED IF STATEMENT IS WHEN ONE IF STATEMENT IS PLACED INSIDE ANOTHER, ALLOWING FOR MULTIPLE CONDITIONAL CHECKS IN A PROGRAM.

```
public class NestedIfExample {
    public static void main(String[] args) {
        int x = 10;
        int y = 5;

        if (x > y) {
            if (x % 2 == 0) {
                System.out.println("x is greater than y and even.");
            }
        }
    }
}
```

Flow of Control: Switch and Loops

- A SWITCH STATEMENT IN JAVA IS A CONTROL FLOW STATEMENT THAT ALLOWS YOU TO EXECUTE DIFFERENT CODE BLOCKS BASED ON THE VALUE OF AN EXPRESSION.

```
int choice = 2;

switch (choice) {
    case 1: System.out.println("You chose option 1"); break;
    case 2: System.out.println("You chose option 2"); break;
    case 3: System.out.println("You chose option 3"); break;
    default: System.out.println("Invalid choice");
}
```

- A FOR LOOP IN JAVA IS A CONTROL FLOW STATEMENT THAT ALLOWS YOU TO REPEATEDLY EXECUTE A BLOCK OF CODE A SPECIFIED NUMBER OF TIMES.

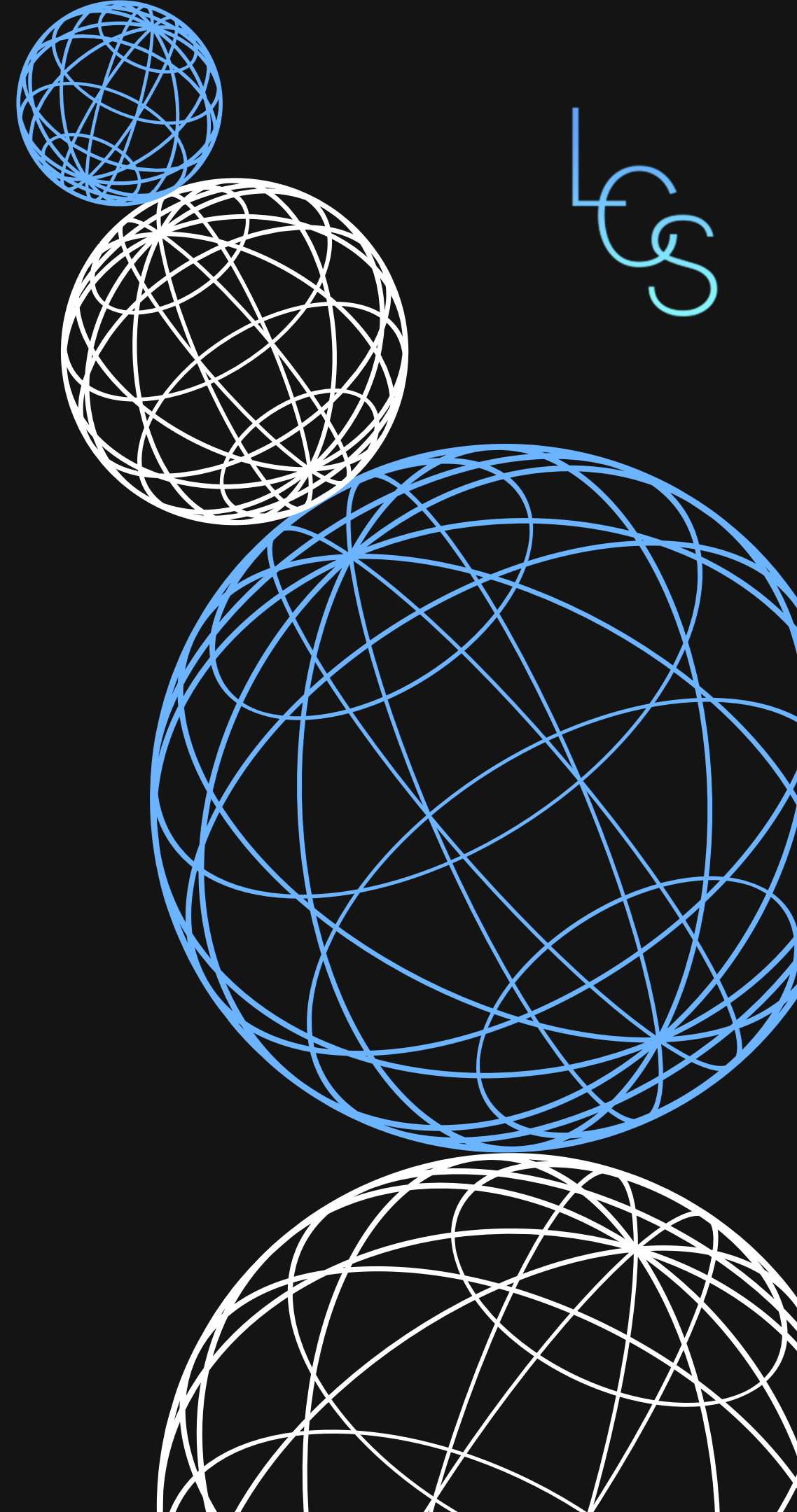
```
for (int i = 0; i < 5; i++) {
    System.out.println("Iteration " + i);
}
```

- A WHILE LOOP IN JAVA ITERATES OVER A BLOCK OF CODE WHILE A SPECIFIED CONDITION IS TRUE

```
int i = 0;
while (i < 5) {
    System.out.println("Iteration " + i);
    i++;
}
```


Do you have any questions?

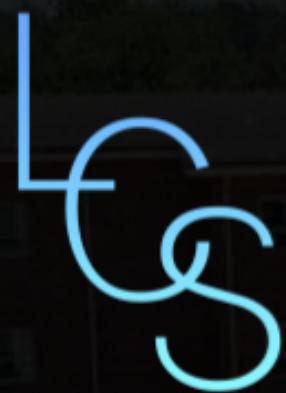
Type them in the chat!!



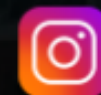
Please fill out this event survey!



Thank you for coming!



Laurier
Computing
Society



@laurier.cs



discord.lauriercs.ca



linktr.ee/lauriercs

