CP213 Midterm 1 Review Session

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

Please take this moment to sign in...

UPCOMING EVENTS...

GitHub 101:

Thursday, October 5th @ 7:00 pm online

Next Review Session:

Tuesday, October 17

Today's Agenda

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

4. ARRAYS

How to work with Java arrays

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

WHAT IS JAVA?

- JAVA IS A CLASS-BASED, OBJECT-ORIENTED PROGRAMMING LANGUAGE
- COMPILED JAVA CODE CAN RUN ON ALL PLATFORMS THAT SUPPORT JAVA WITHOUT THE NEED FOR RECOMPILATION.
- IT WAS CREATED BY SUN MICROSYSTEMS TEAM LED BY JAMES GOSLING IN 1991 (NOW OWNED BY ORACLE)
- JAVA IS A HIGH LEVEL LANGUAGE

IMPLEMENTATION OF A JAVA APPLICATION

1. CREATING THE PROGRAM

- WRITE YOUR PROGRAM CODE WITHIN THE MAIN() METHOD

2. COMPILING THE PROGRAM

- TO COMPILE THE PROGRAM, RUN THE JAVA COMPILER, WITH THE NAME OF THE SOURCE FILE ON COMMAND PROMPT LIKE AS FOLLOWS
  - JAVAC SOURCEFILENAME.JAVA --> .CLASS
  - COMPILES INTO A BINARY BYTE CODE USING JAVA COMPILER

3. RUNNING THE PROGRAM

- THAT BYTE CODE RUNS ON THE JVM (JAVA VIRTUAL MACHINE), WHICH IS A SOFTWARE BASED INTERPRETER

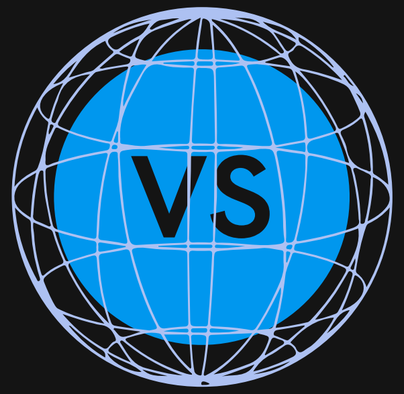
Java Primitive Data Types

TYPE NAME	KIND OF VALUE	SIZE RANGE
boolean	true or false	not applicable
char	single character (Unicode)	all Unicode characters
byte	integer	-128 to 127
short	integer	-32768 to 32767
int	integer	-2147483648 to 2147483647
long	integer	-9223372036854775808 to 9223372036854775807
float	floating-point number	-3.40282347 x 10^+38 to -1.42039846 x 10^-45
double	floating-point number	+- 1.76769313486231570 x 10^+308 to +-4.94065645841246544 x 10^-324

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.



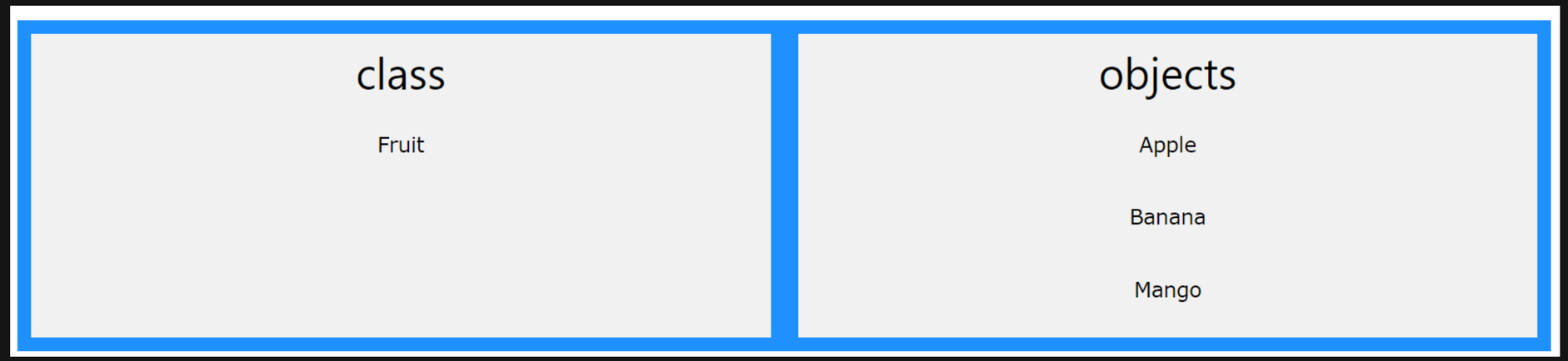
OBJECT ORIENTATED PROGRAMMING

- 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:



Do you have any questions?

Console I/O: Input

```java
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();
    }
}
```

Package: a grouping of classes and interfaces in a file system which work together

The + signs are for string concatenation

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:

ReviewSession.java

```java
package cp213;

public class ReviewSession {
    public static void main(String[] args) {
        System.out.println("THE ANSWER IS " + 42);
    }
}
```

Console

```
<terminated> ReviewSession [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe  (Sep. 12, 2023, 8:09:27 p.m. – 8:09:28 p.m.) [pid: 3272]
THE ANSWER IS 42
```

Do you have any questions?

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.

```java
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.");
            }
        }
    }
```

Output:

```
"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.

```java
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");
}
```

Output:

```
You chose option 2
```

- 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.

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

Output:

```
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
```

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

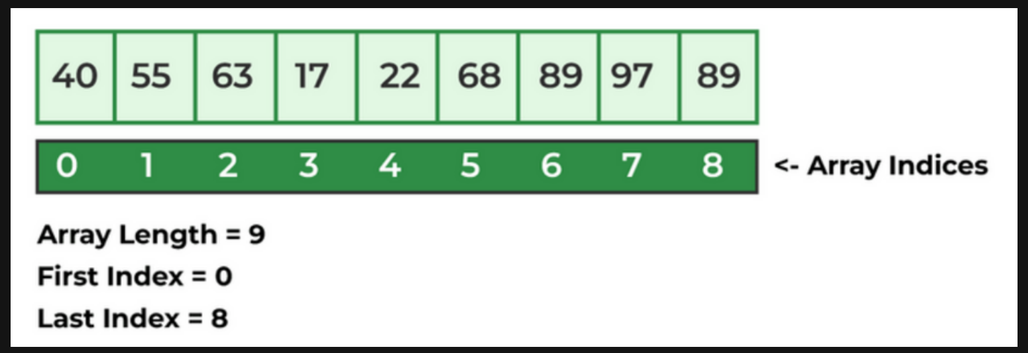
```java
int i = 0;
while (i < 5) {
    System.out.println("Iteration " + i);
    i++;
}
```

Output:

```
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
```

Arrays: Introduction

- AN ARRAY IS A DATA STRUCTURE USED TO PROCESS A COLLECTION OF DATA THAT IS ALL OF THE SAME TYPE



- AN ARRAY THAT BEHAVES LIKE THIS COLLECTION OF VARIABLES, ALL OF TYPE **DOUBLE**, CAN BE CREATED USING ONE STATEMENT AS FOLLOWS:
- **DOUBLE[] SCORE = NEW DOUBLE[5];**
- **OR USING TWO STATEMENTS:**
  - **DOUBLE[] SCORE;**
  - **SCORE = NEW DOUBLE[5];**

Arrays Cont'd

- THE INDIVIDUAL VARIABLES THAT TOGETHER MAKE UP THE ARRAY ARE CALLED INDEXED VARIABLES
- IN JAVA, INDICES MUST BE NUMBERED STARTING WITH **0**, AND NOTHING ELSE
    - EX: **SCORE[0], SCORE[1], SCORE[2], SCORE[3], SCORE[4]**

- JAVA ARRAYS ARE OBJECTS!!!
    - NOT ALL LANGUAGES HAVE ARRAYS AS OBJECTS
- AN ARRAY HAS ITS LENGTH FIXED WHEN INITIALISED
- LIKE PYTHON, JAVA ARRAYS CAN BE MULTIDIMENSIONAL

Arrays: Example

long form

```
double[] a;
a = new double[N];
for (int i = 0; i < N; i++)
   a[i] = 0.0;
```

declaration

creation

initialization

short form

```
double[] a = new double[N];
```

initializing declaration

```
int[] a = { 1, 1, 2, 3, 5, 8 };
```

ArrayList

- ARRAYLIST: A JAVA DATA STRUCTURE AND OBJECT WHICH ACTS LIKE AN ARRAY WITH A DYNAMIC (CAN BE CHANGED) LENGTH
- ARRAYLIST<STRING> REPRESENTS ARRAYLIST OF STRINGS.
- ARRAYLIST<TYPE> LIST = NEW ARRAYLIST<TYPE>();
    - INITIALISE
- METHODS DEMONSTRATED ON THE NEXT SLIDE

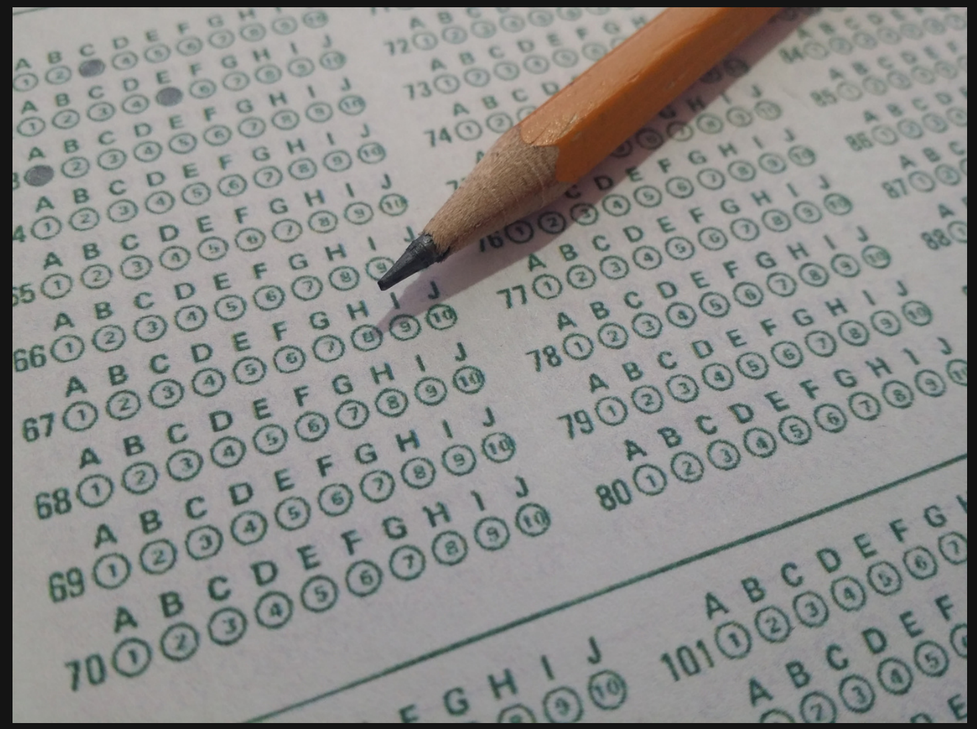
ArrayList Methods

- LIST.ADD(ELEMENT);
  - THIS LETS YOU ADD THE ELEMENT TO THE ARRAYLIST
- TYPE ELEMENT = LIST.GET(INDEX);
  - YOU CAN ACCESS ELEMENTS IN THE ARRAYLIST BY THEIR INDEX USING THE **GET** METHOD
- INT SIZE = LIST.SIZE();
  - RETURNS SIZE OF ARRAYLIST
- REMOVE METHOD
  - LIST.REMOVE(INDEX); TO REMOVE BY INDEX
  - LIST.REMOVE(ELEMENT); TO REMOVE AN ELEMENT

Do you have any questions?

Kahoot time!!!!1!!

Good luck on the midterm!!!!



Please fill out this event survey!

Thank you for coming!

Laurier
Computing
Society

@laurier.cs

discord.lauriercs.ca

linktr.ee/lauriercs