CS151 Intro to Data Structures

Java Basics

Administrivia

- Course website
  - BMC-CS-151.github.io
  - Assignments and lab instructions, syllabus
- Piazza:
  - Asynchronous communication
  - Can post anonymously (anonymous just to classmates)
  - Answer your peers questions!
    - Counts for participation grade
- Gradescope:
  - Submit all assignments
  - Can request re-grade requests

Schedule

- Assignments due on Thursdays released on Sundays
  - 20 points deducted each day. After two days, the submission window will be closed.
- Lab Park 231/M 2:40pm-4:00pm (After class)
  - Attendance required
- Midterm: March 4th (Wednesday before Spring break)
- Final Exam: self scheduled

AI Disclaimer!

Syallbus

- Homeworks: 35%
- Labs: 5%
- Midterm: 20%
- Final: 35%
- Participation: 5%

Course Staff

- Ruth Tilahun
- Reagan Buvens
- Clara Fee
- Khahn Ha Nguyen
- Renata Del Vecchio

Course Staff

- Office hours Park 231:
  - Monday 8-10pm
  - Tuesday 8-10pm
  - Wednesday 6-10pm
  - Thursday 6-10pm
  - Friday 10-12am (professor)

Dr. Elizabeth Dinella

- 1st year at BMC
- Recent Penn Grad: (PhD thesis neural inference of program specifications)
- Office Hours: Friday 10-11am (zoom)
- Research:
  - Program Analysis
  - Machine Learning
  - Web3 Security

First Things

- CS server account
  - Make sure you can log in
  - Email David Diaz if encountering issues (ddiaz1@brynmawr.edu)

- Lab00: ideally completed already, getting up and running with vim and linux

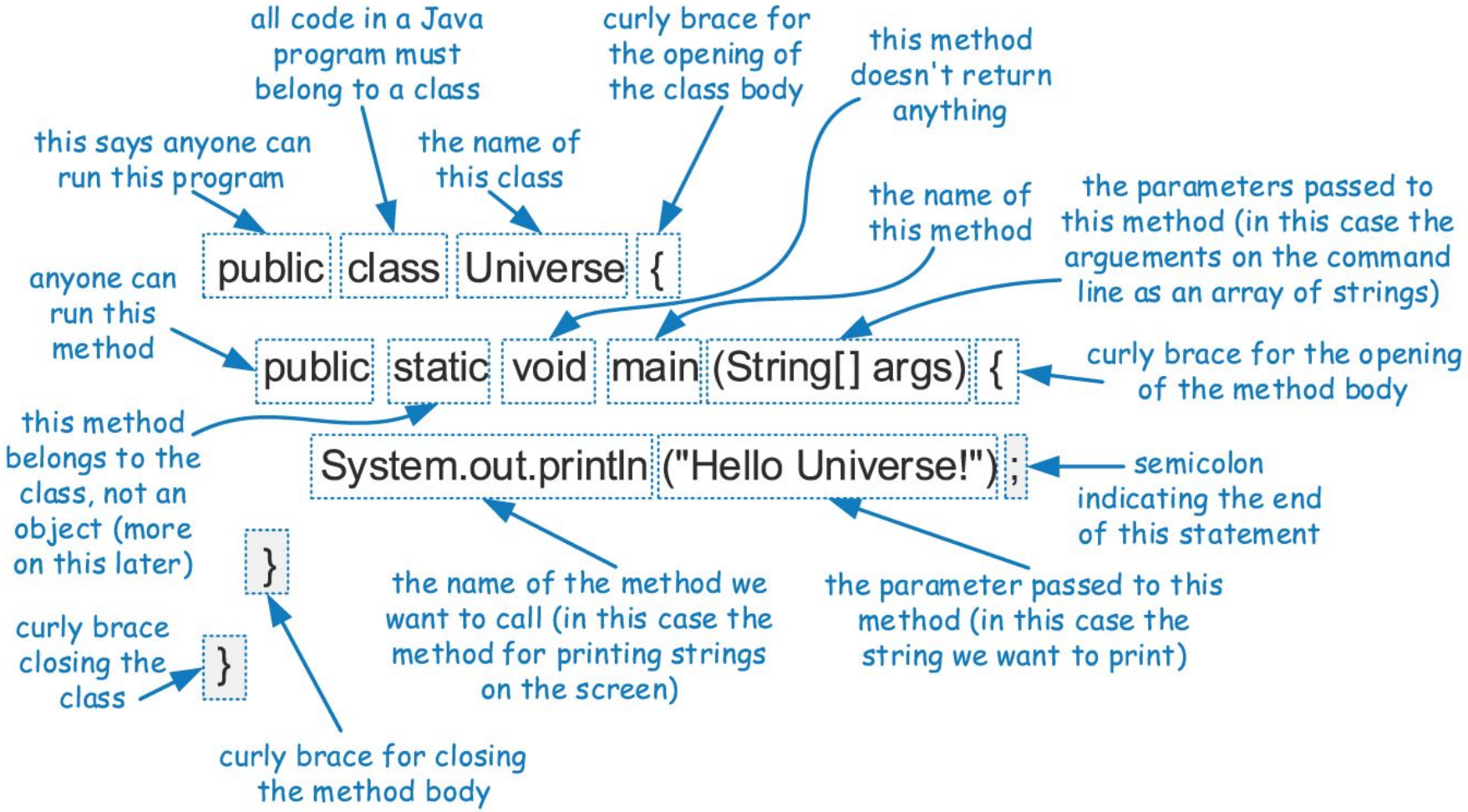
- Lab attendance is required.

- Software: vim, Java, or just ssh

Outline

- Data Types
- Objects
- String review
- Input (Scanner)
- OOP (Inheritance)
- File I/O, Exceptions
- ***Not reviewing:***
  - Methods
  - Loops

An Example Program



Java: A compiled language

- Java program in .java (source code)
- Compiler create .class file (byte code)
- Java Virtual Machine (JVM) execute the code

Java Basics

- Name of main class and file must agree
  - `class Driver` <--> `Driver.java`
- Compilation
  - `javac Driver.java`
- Execution
  - `java Driver`

Components of a Java Program

- Statements are placed in *methods*, that belong to class definitions.
- The static method named `main` is the first method to be executed when running a Java program.
- Any set of statements between the braces `{` and `}` define a program block.

Base/Primitive Types

- Variables must have types
  - base type
- Types define memory used to store the data

- Primitives:

boolean	a boolean value: true or false
char	16-bit Unicode character
byte	8-bit signed two's complement integer
short	16-bit signed two's complement integer
int	32-bit signed two's complement integer
long	64-bit signed two's complement integer
float	32-bit floating-point number (IEEE 754-1985)
double	64-bit floating-point number (IEEE 754-1985)

```
boolean flag = true;
boolean verbose, debug;
char grade = 'A';
byte b = 12;
short s = 24;
int i, j, k = 257;
long l = 890L;
float pi = 3.1416F;
double e = 2.71828, a = 6.022e23;
```

Classes and Objects

- ***Classes*** are blueprints, ***objects*** are instance of the classes

- A class defines:
  - instance variables – what the object stores
  - Methods – how the object functions

- Every variable is either a primitive or a reference to an object

Counter Example

What are the consequences of Objects as references?

- Simple Counter DEMO

More Complex Counter Example

Let’s code!

Access Control Modifiers

- `public:`
  - designates that all classes may access
- `private:`
  - designates that access is granted only to code within that class.
- protected:
  - child classes may access
- `static`
  - associates a variable/method with the class as a whole, rather than with each individual instance of that class

javadoc comments

- Comments
  - /* */
  - //
- A style/format of commenting for auto-generation of documentation in html

  /**

  */
  - used for method headers and classes

Example

```
/**
 * returns the sum of two integers
 * @param x The first integer
 * @param y The second integer
 * @return int The sum of x+y
 */
int sum(int x, int y)
```

Casting – convert the type

- **More coding :)**

Equality - More coding :)

`String` class methods

- `charAt(int index)`
  - Returns the character at the specified index
- `equals(String anotherString)`
  - Compares a string to a specified object
- `indexOf(char c)`
  - Returns the index value of the first occurrence of a character within the input string
- `indexOf(String str)`
  - Returns the index value of the first occurrence of a substring within the input string
- `length()`
  - Returns the number of characters in the input string
- `substring(int startIndex, int endIndex)`
  - Returns a new string that is part of the input string
- `toLowerCase()`
  - Converts all the characters to lower case
- `toUpperCase()`
  - Converts all the characters to upper case
- `String concat(String anotherString)`
  - Concatenates with anotherString and returns it

Parsing!

- Coding!

How do we Output?

- `System.out.println()`
- `System.out.print()`

How do we take Input?

More code!

Exceptions – way to deal with unexpected events during execution

- Unexpected events:
  - unavailable resource
  - unexpected input
  - NPE
  - AOB

How do we deal with exceptions?

```
try {
    guardedBody
} catch (exceptionType1  variable1) {
    remedyBody1
} catch (exceptionType2  variable2) {
    remedyBody2
} ...
    ...
```

How do we deal with exceptions?

Back to our example code!

Printing Objects

code!

What you should know/review

- variables
- expressions
- operators
- methods
  - parameters
  - return value
- conditionals
- `for`/`while` loops
- class design and object construction
  - instance variables
  - constructor
  - getters/setters
  - class methods
  - `new`
- arrays
- arrays of objects
- `String`

What you don’t know

- Read the manuals/references
  - Unix commands (flags, usage, examples)
  - Java methods (parameters/overloading)
- Google – but with judgement
- **AI Disclaimer**
- Trial-and-Error is a fundamental method of problem-solving
- The ability to tinker is a fundamental engineering/CS skill