CS171 Introduction to Computer Science II
Spring 2023

Today: Course Overview & Logistics
(Time permitting: Java Review Part1)

What is this course about?

- Continuation of CS170
  - Use CS170 skills to solve bigger problems / real-world applications!

- Key differences with CS170
  - CS170 focused on the “how” of programming, CS171 helps you understand the “why”, “what” & “where” to apply these skills
  - No Lab! (Less programming in class compared to CS170)

What is this course about?

Hardware vs. Programmers:

Which, do you think, is relatively more expensive?

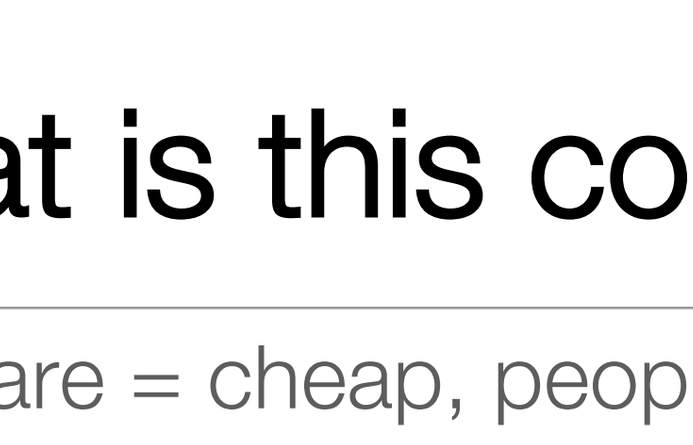
What is this course about?

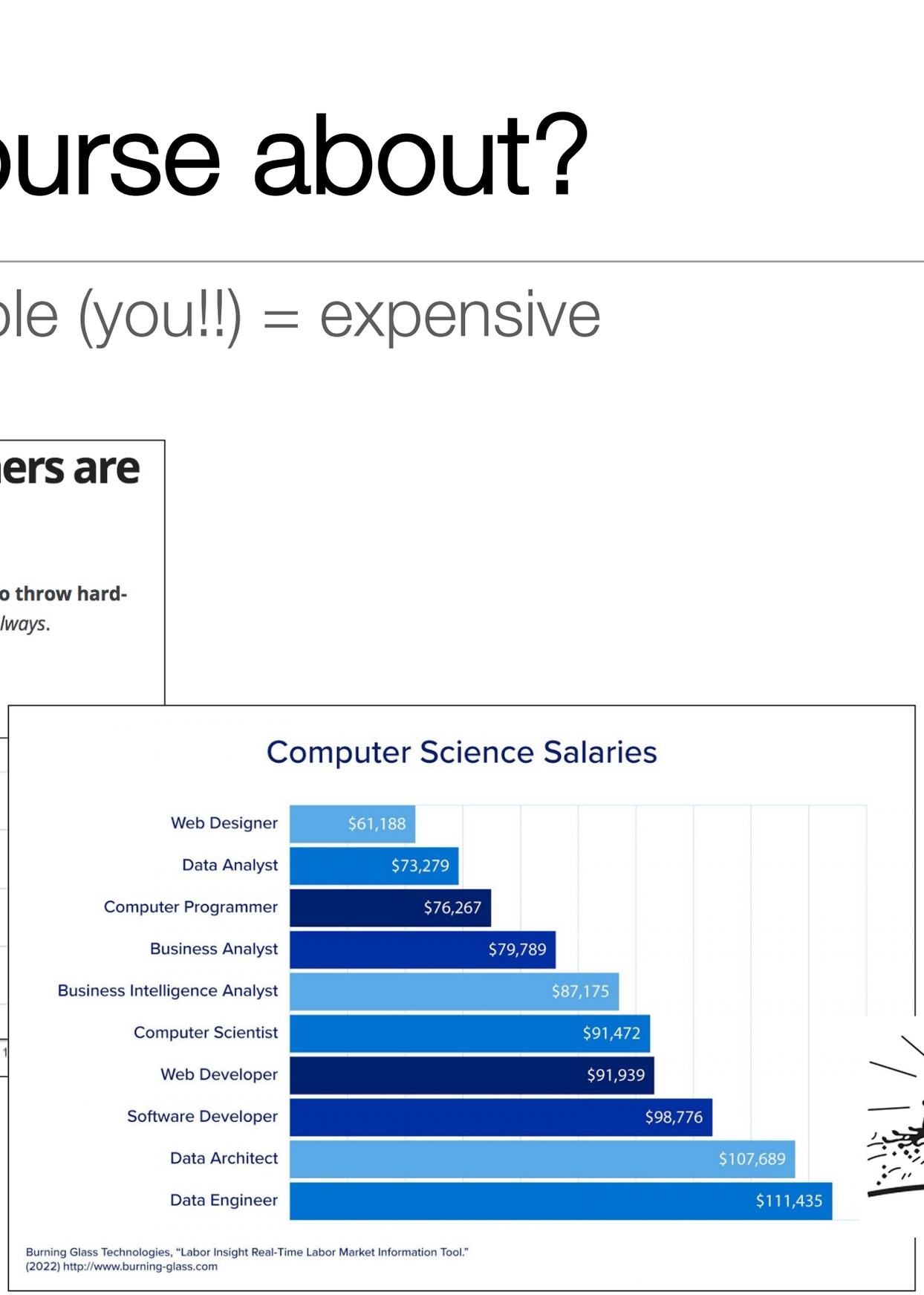
Hardware = cheap, people (you!!) = expensive

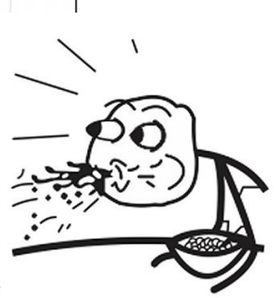
Hardware is Cheap, Programmers are Expensive

Given the rapid advance of Moore's Law, **when does it make sense to throw hardware at a programming problem?** As a general rule, I'd say almost *always*.

Consider the average programmer salary here in the US:







Introductions

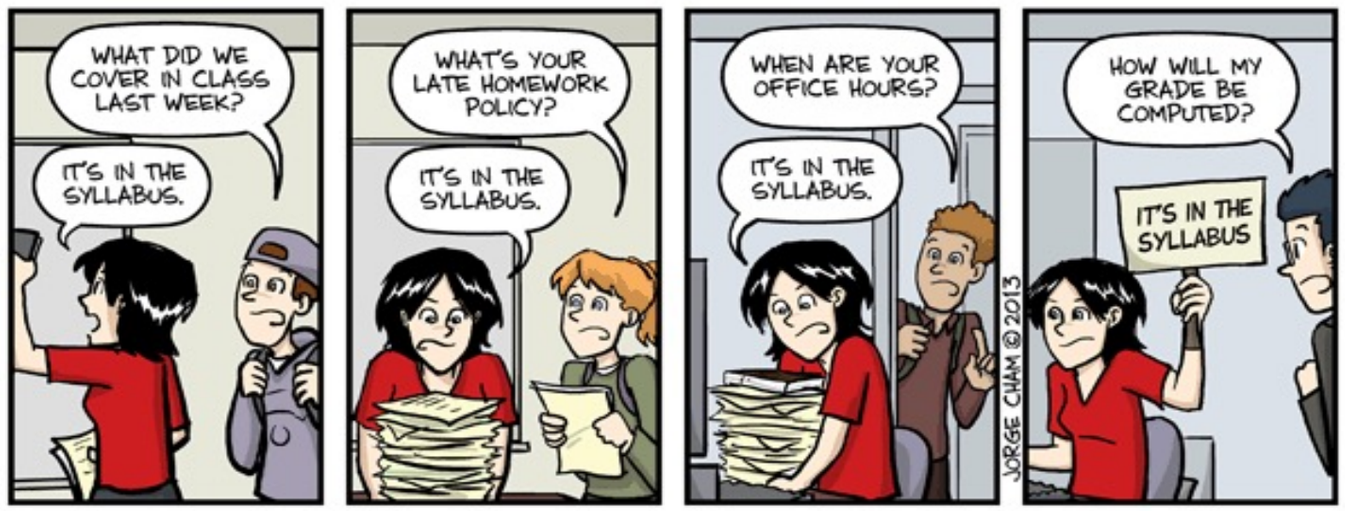
- Instructor: Nosayba El-Sayed
  - I'm a teaching professor in CS-Emory since 2018
  - Before that: researcher at MIT; instructor and graduate researcher at Univ of Toronto; research intern at Amazon
  - Contact info available on our Canvas *Syllabus* page

- TAs: TBD
- Full office hours schedule will be updated in Syllabus (check Canvas regularly!)
  - A survey will follow (after 1~2 weeks) to make sure our O.H. schedule works well for you all, and we'll make changes if needed :-)

Introductions

- How about you all? :-)
- Show of hands:
  - first-year students?
  - (intended) major: CS?
  - (intended) major: Economics?
  - (intended) major: Math/Applied Math?
  - (intended) major: Biology or Chemistry or Physics?
  - (intended) major: Business Admin.?
  - (intended) major: other (e.g. psychology, sociology, history, ..)
  - joining us from Oxford?

More course logistics…



IT'S IN THE SYLLABUS

This message brought to you by every instructor that ever lived.

WWW.PHDCOMICS.COM

Communication & Platforms

1. Canvas

   - Syllabus + tentative schedule
   - Modules: Material grouped by lecture date
   - Weekly online quizzes

2. Gradescope: Programming Assignments
(you'll be auto-enrolled using your NetID, no need to worry about self-enrolling)

3. Piazza (you will be auto-enrolled by end of this week):
https://piazza.com/emory/spring2023/cs171

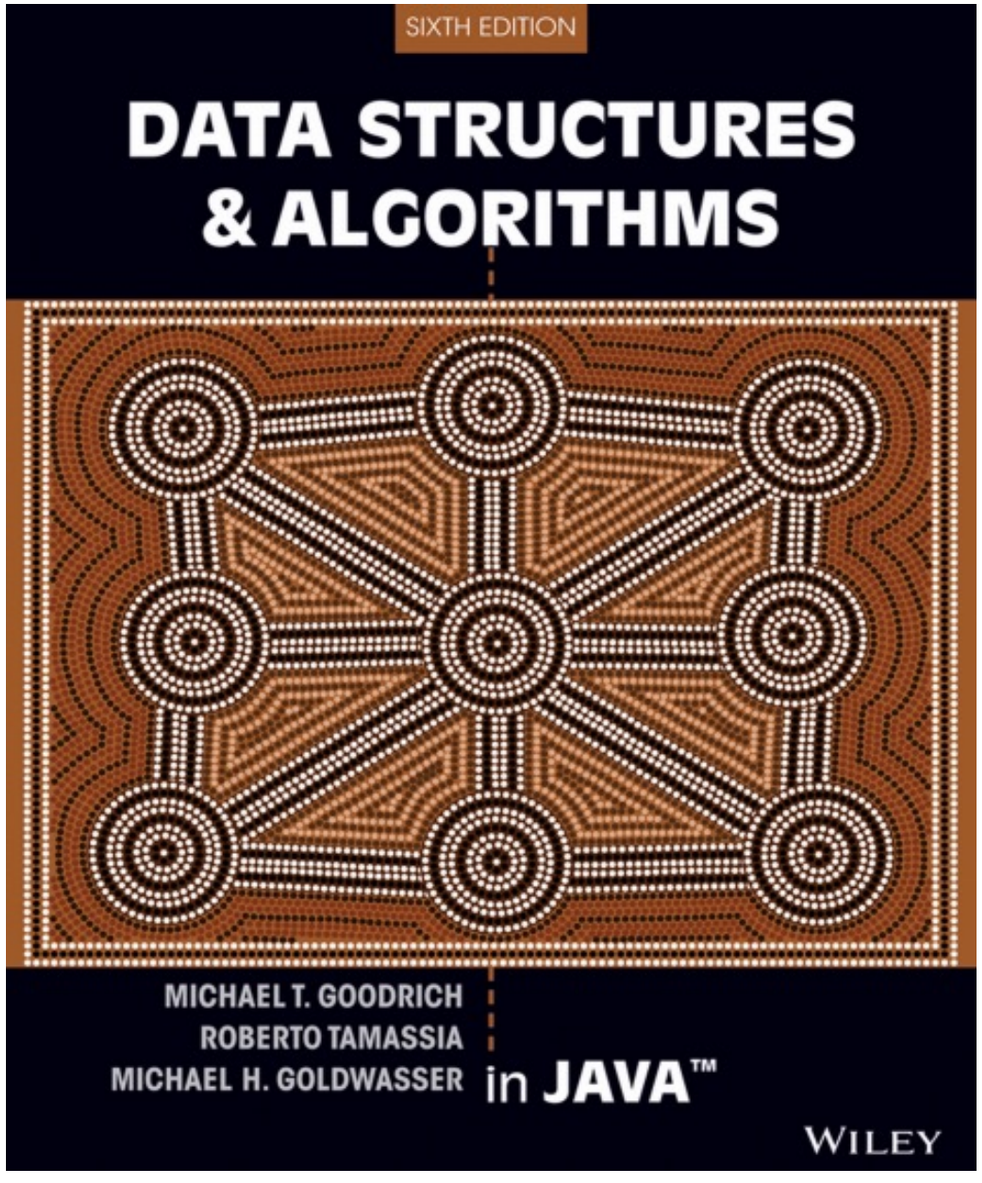
   - All questions related to course material, assignments, quizzes, etc. should be posted on the **public** forum on Piazza :-)

Lectures

- Lecture slides cover essential material
  - Best reference for the course
  - Book may provide more details and the appropriate chapters are noted in the book
- In-class exercises (exercise worksheets or programming)
  ➔ improve your learning experience significantly!

Course Textbook

- Data Structures and Algorithms in Java (6th Edition) by Goodrich et al.
- Website: https://www.wiley.com/en-us/Data+Structures+and+Algorithms+in+Java%2C+6th+Edition-p-9781118771334
- Amazon link: https://www.amazon.com/Data-Structures-Algorithms-Java-6th-ebook/dp/B00JDRQF8C
- Optional reading material will be noted on the course website or slides



Class Expectations

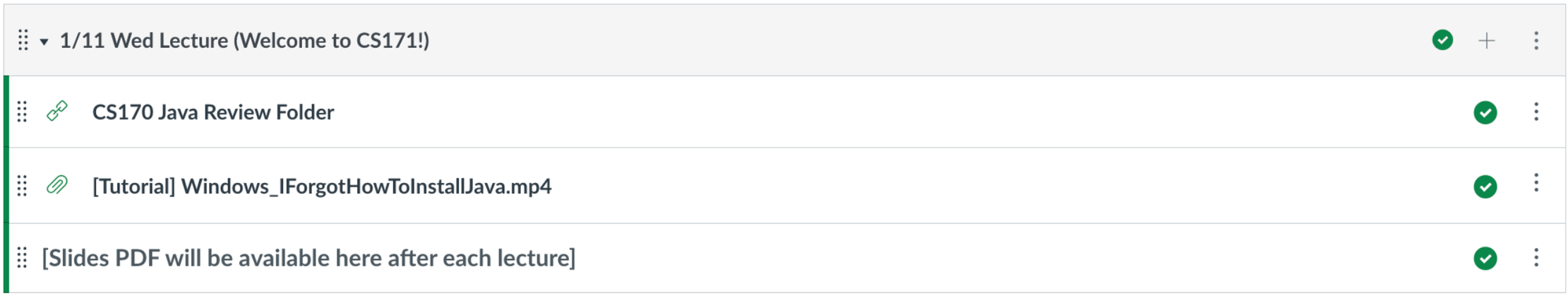
- Programming assignments (40%)
- Test 1, Mid semester (20%)
- Test 2, End of semester (25%)
- Online Quizzes (15%)

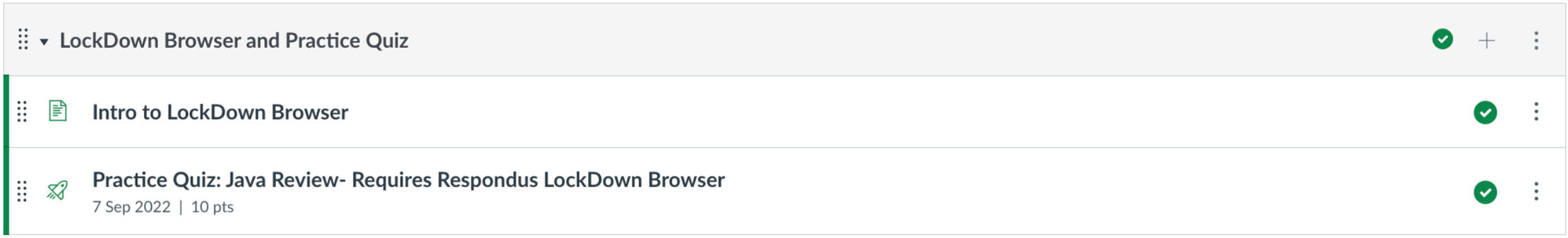
  ➔ Weekly short Canvas quizzes, requires **LockDown** browser;
  we will drop your worst quiz grade eventually

  ➔ Drop-lowest policy is intended to accommodate for any circumstances (illness, emergencies, etc.)

  ➔ For extreme circumstances (e.g., missing 2 consecutive quizzes) => Contact me in advance and submit proper form to Office of Undergrad Education

Download LockDown browser (see practice quiz under **Modules**)





Assignments: Honor Code

- College Honor Code and Departmental Policy
- Acceptable and encouraged to discuss high-level concepts with other students but **ANY CODE YOU SUBMIT MUST BE YOUR OWN**
- We use software that detects plagiarism
  - Past cases in CS171 were investigated and students were failed the class (last year a student was dismissed entirely from Emory University…) :-(
- All assignments must include the following comment at the top of the file:

```
/*
THIS CODE IS MY OWN WORK, IT WAS WRITTEN WITHOUT
CONSULTING CODE WRITTEN BY OTHER STUDENTS OR COPIED
FROM ONLINE RESOURCES . _Your_Name_Here_
*/
```

Policies

- Late Assignments
  - Each student has one 24-hour "late day" token that can be used on any of the assignments!
  - After you've used your token, assignments will still be accepted up to 24 hours late, but with a 10% penalty (automatically deducted)
  - Assignments turned in more than 24 hours late will not be accepted
  - Additional extensions on assignments will be granted with appropriate documentation from OUE
- Tests cannot be rescheduled without a letter from OUE

CS171 Success Recipe

- Attend **lectures**, think **critically**, and be **active**
- Check **Syllabus/Modules** on Canvas regularly (pointers to lecture PDFs, exercise solutions, assignment dates, etc.)
- Come to **office hours** (TAs and me)
- Do **assignments** — start early!

- Study for **exams**
- **Have fun!!**

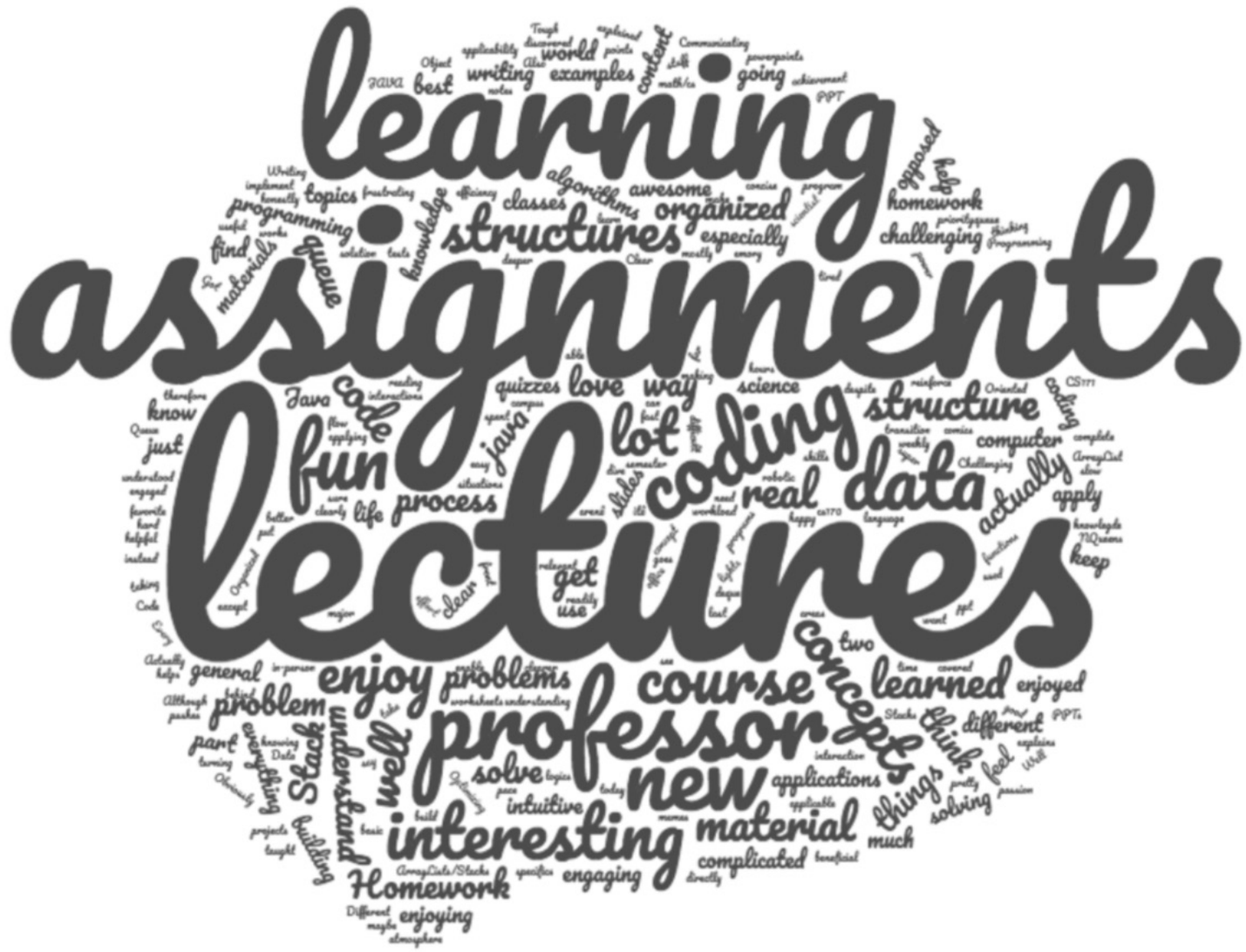
Success Recipe, Contd.: Self Care

- Personal and academic stressors may create barriers to learning
- Take time out of your busy schedule to do something that relaxes you
- See course syllabus for university resources that are available to support you

Past Student Feedback

What is your most favorite part of CS171?

A word cloud of 100+ answers



Now back to course content…

Course Content

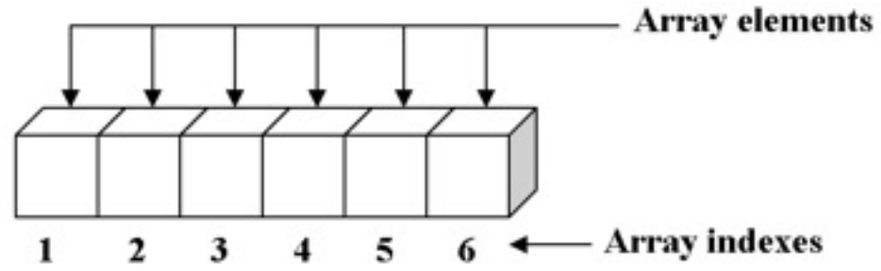
- Continuation of CS 170
- Programming / problem solving with code

Data Structures:
Store and manage information

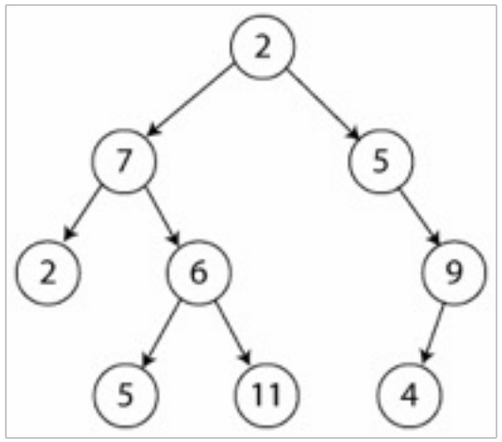
Algorithms:
Steps (recipe) to solve problems

Data Structure

- Logical construct for organizing and accessing information
- Different data structures are useful for different types of problems
- Analogy: Cars & boats



One-dimensional array with six elements



Tree with nine elements

Why?

“I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. ***Bad programmers worry about the code. Good programmers worry about data structures and their relationships.***

”

— Linus Torvalds (creator of Linux)

What is an “Algorithm”

- **Algorithm** is a method for solving a problem expressed as a sequence of steps
  => suitable for execution by a computer
- Can be expressed in different forms: natural language, flowcharts, pseudocode, programming languages

Muhammad
Al-Khwarizmi

Algorithms are Everywhere

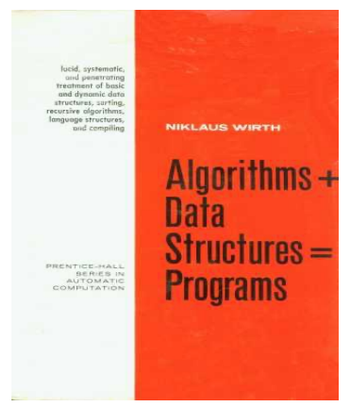
Is it “Good”?

- How do you measure the “**goodness**” of data structure and algorithms?
  - **Speed** - how quickly will I get my answer?
  - **Size** - how much disk / memory / network will it use?
  - **Flexibility/scalability** - how will it work if I get 1 million new entries?

“Program”

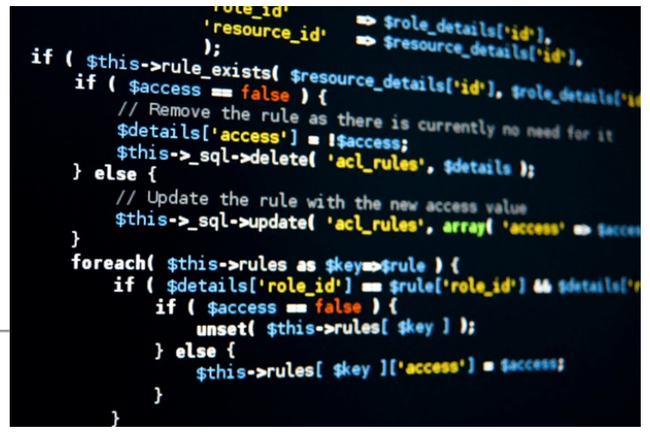
“ Algorithms + Data Structures = Programs. ”

— Niklaus Wirth



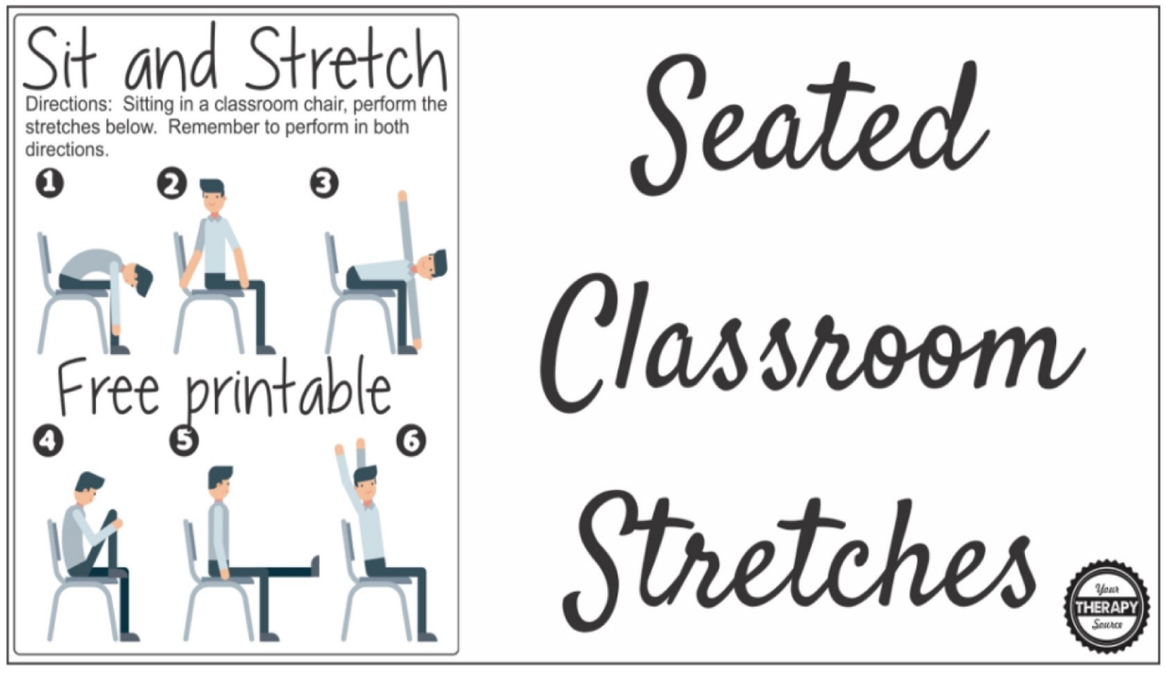
- **Algorithm** must use some **data structure** to store its information
- **Algorithm** manipulates data in the data structure in various ways

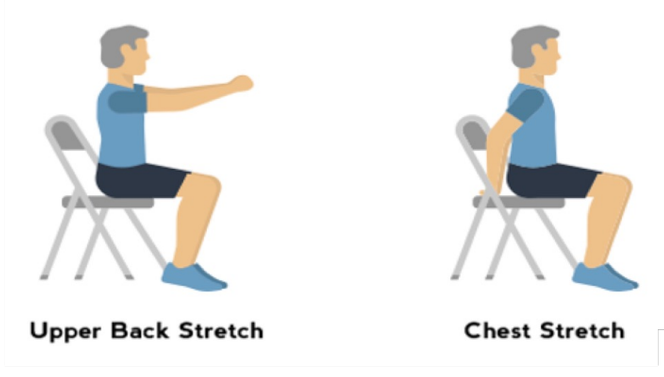
Writing a Program



- Determine your **problem** (requirements)
- Design **data structures** to store information
- Design **algorithm** that uses information to solve the problem
- **Implement** the algorithm
- **Test/debug** your code

(Quick;-) Stretch Break





Java Review

CS 171: Introduction to Computer Science II

Review: Java Data Types

```
double pi = 3.14;
```

What does a type determine?

- range of values var can hold;
- operations supported;
- meaning of operations!

(example: "+" for int is different than "+" for String)

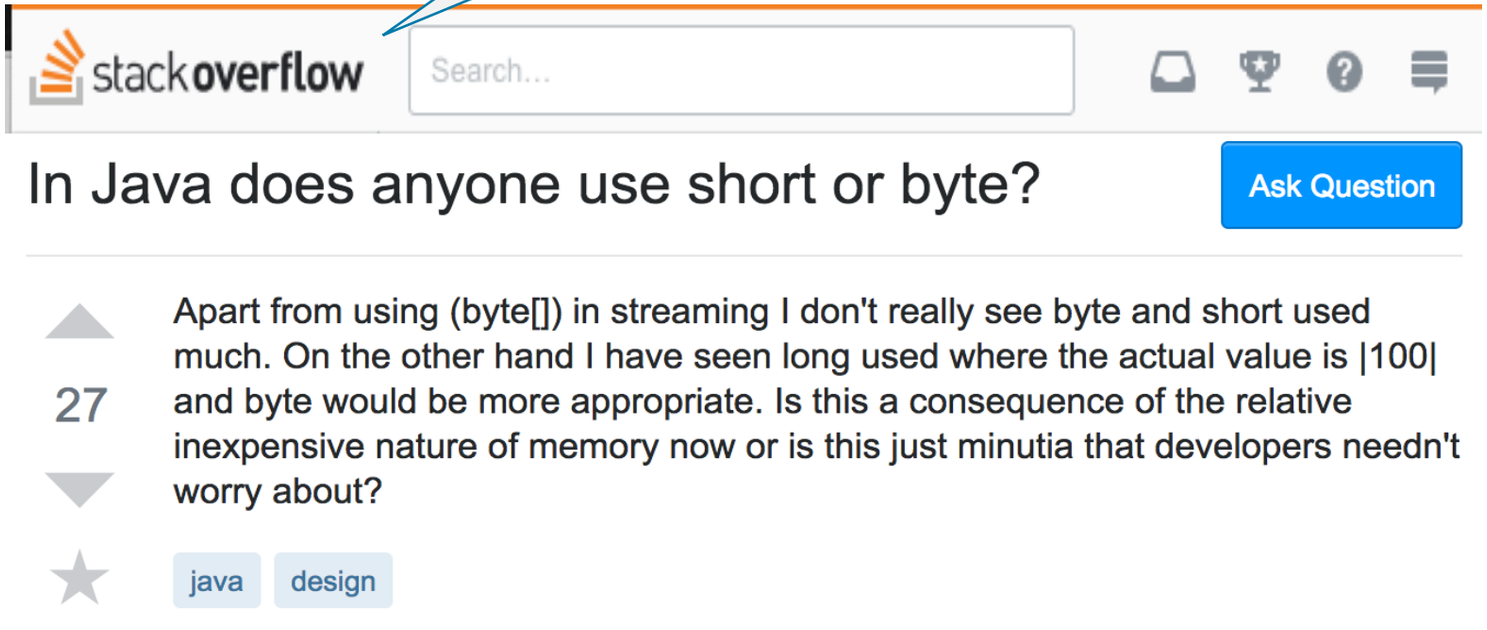
Java Review: Primitive Data Types

- Integers with arithmetic operations:
  `byte`, `short`, `int`, `long`
  `8bits  16bits  32bits  64bits`

  How much memory is reserved?

Mobile apps

StackOverflow community is a great resource for programmers



Java Review: Primitive Data Types

- Integers with arithmetic operations:
  `byte`, `short`, `int`, `long`

- Real numbers with arithmetic operations:
  `float`, `double`

```
double pi = 3.14;
```

- Boolean operations with {`true`, `false`} values and logical operations: `boolean`

- Characters: `char`

Java Review: Variables

- *Variable* is a name for a location in memory used to hold a data value; has:
  - type, name (identifier), and value

```
double pi = 3.14;
```

1) Declaration: type and name

2) Assign value to variable *(may be done in same declaration statement)*

3) Use a variable in an expression

- Variable cannot be used if it is not declared or initialized
- Left hand side of assignment operator is always a variable and right hand side is an expression

Java Review: Variable Usage

Declaration: Instructs the compiler to reserve a portion of main memory to hold a particular value referred by a particular name

```
int a, b, c;           // Declares three ints, a, b, and c.
int a = 10, b = 10;    // Example of initialization
byte B = 22;           // initializes a byte type variable B.
double pi = 3.14159; // declares and assigns a value of PI.
char a = 'a';          // the char variable a is initialized with value 'a
```

https://www.tutorialspoint.com/java/java_variable_types.htm

Java ~~Review~~: Object Data Types

- Pretty much everything else!
- `array` types
- Class types
  - Example of a class type we used a lot in CS170: `String`
- Interface types

Java Review: Expressions

- An expression is a combination of one or more operators and operands that perform a calculation
  - Operands might be numbers, variables, or expressions

- Arithmetic expressions
  - Ex: `int score = x - 10 * lateDays;`

- Boolean expressions
  - Ex: `boolean isLate = submissionDate <= dueDate;`

`true or false`

Java Review: Comparison Operators

Operator	Name
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
!=	not equal to

Java Review: Common Operators

type	set of values	operators	typical expressions	
			expression	value
`int`	integers between -2^{31} and $+2^{31}-1$ (32-bit two's complement)	+ (add)	`5 + 3`	8
		- (subtract)	`5 - 3`	2
		* (multiply)	`5 * 3`	15
		/ (divide)	`5 / 3`	1
		% (remainder)	`5 % 3`	2
`double`	double-precision real numbers (64-bit IEEE 754 standard)	+ (add)	`3.141 - .03`	3.111
		- (subtract)	`2.0 - 2.0e-7`	1.9999998
		* (multiply)	`100 * .015`	1.5
		/ (divide)	`6.02e23 / 2.0`	3.01e23
`boolean`	`true` or `false`	&& (and)	`true && false`	false
		|| (or)	`false || true`	true
		! (not)	`!false`	true
		^ (xor)	`true ^ true`	false
`char`	characters (16-bit)	*[arithmetic operations, rarely used]*		

Java Review: Assignment Operators

Operator	Example	Equivalent Code
+=	i += 8	i = i + 8
-=	i -= 8	i = i - 8
*=	i *= 8	i = i * 8
/=	i /= 8	i = i / 8
%=	i %= 8	i = i % 8

Java Review: Pre/Post Operators

Operator	Name	Description
++var	preincrement	Increments var by 1 and evaluates to the new value after the increment
var++	postincrement	Evaluates to the original value and increments var by 1
--var	predecrement	Decrements var by 1 and evaluates to the new value after the increment
var--	postdecrement	Evaluates to the original value and decrements var by 1

Java Review: Increment Operators

```java
int i = 10;
int newNum = 10 * (i++);
```

```java
int newNum = 10 * i;
i = i + 1;
```

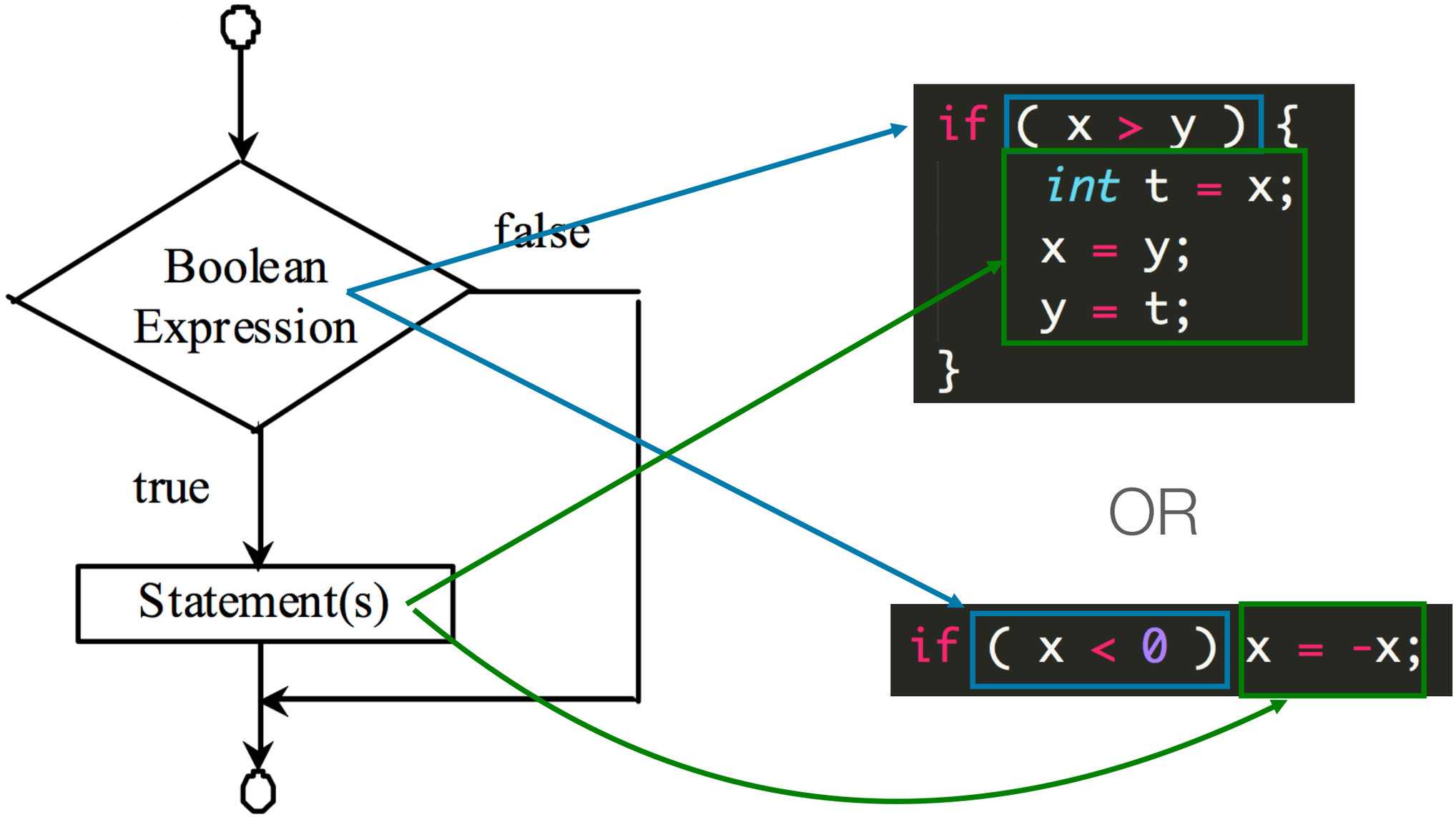
```java
int i = 10;
int newNum = 10 * (++i);
```

```java
i = i + 1;
int newNum = 10 * i;
```

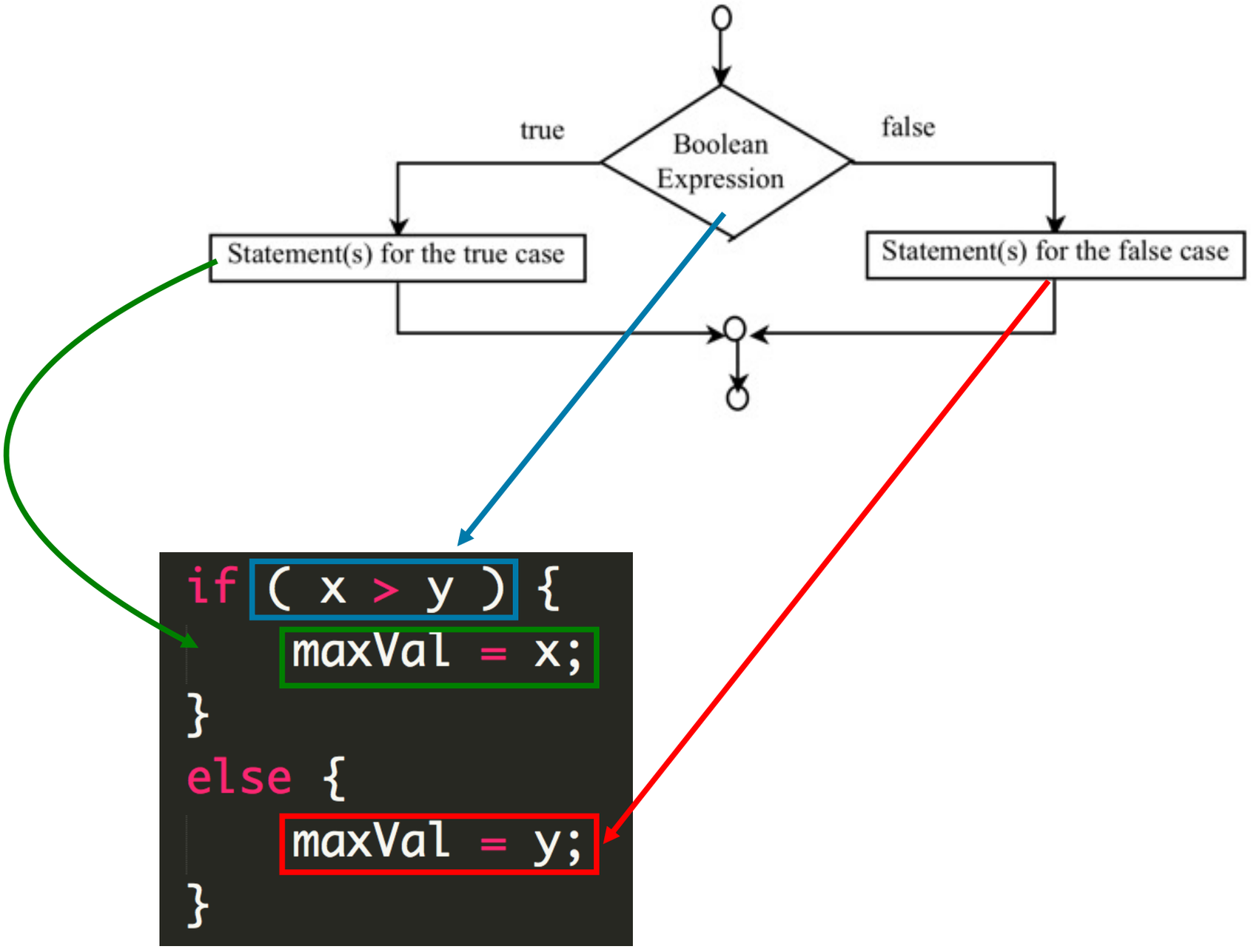
Java Review: Control Flow

- Conditionals
- Loops
- Break and continue

Java Review: If Statement



Java Review: If-Else Statement

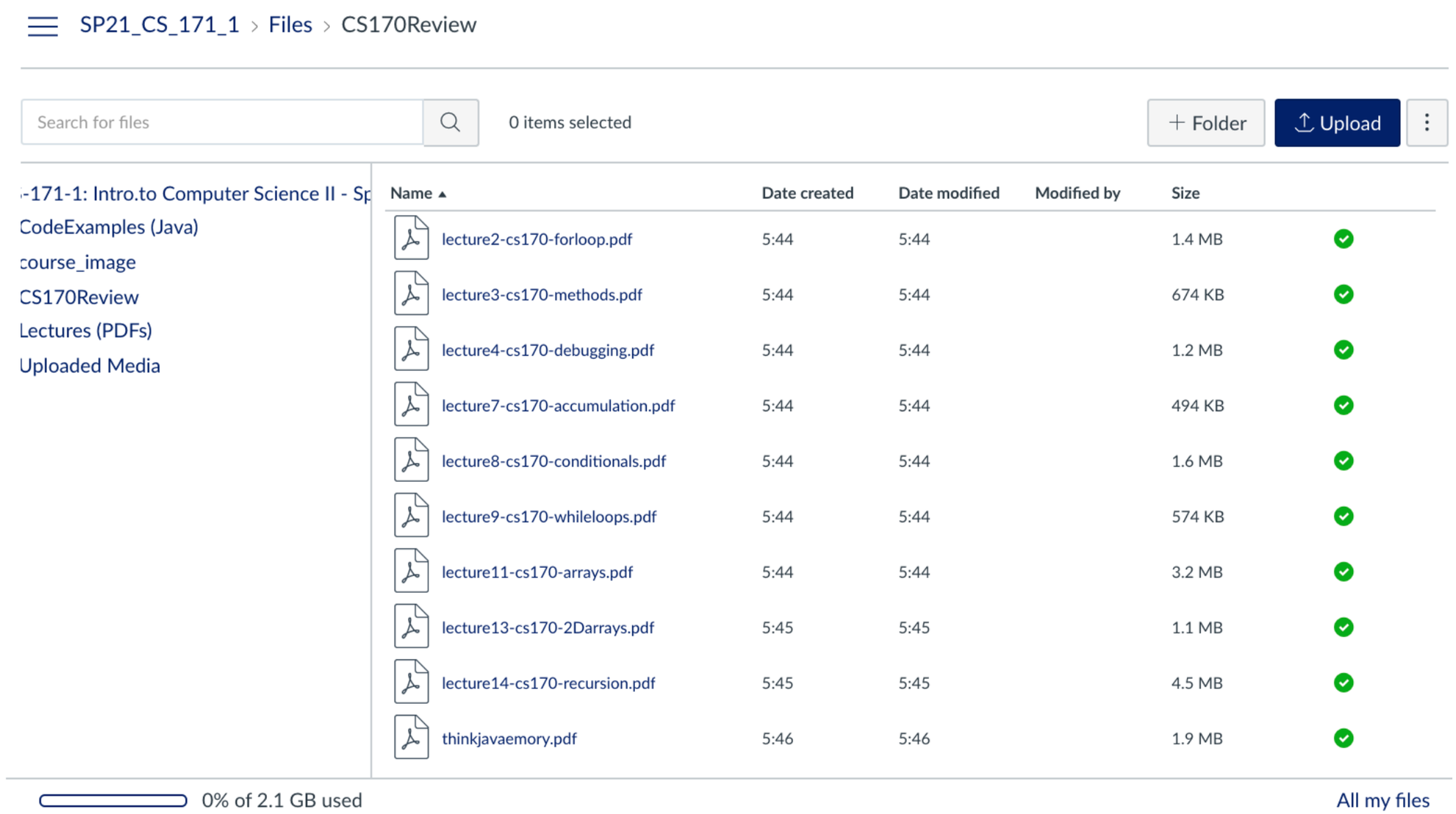


Java Review: Nested If-Else

```
if      (income <     0) rate = 0.00;
else if (income <  8925) rate = 0.10;
else if (income <  36250) rate = 0.15;
else if (income <  87850) rate = 0.23;
else if (income < 183250) rate = 0.28;
else if (income < 398350) rate = 0.33;
else if (income < 400000) rate = 0.35;
else                      rate = 0.396;
```

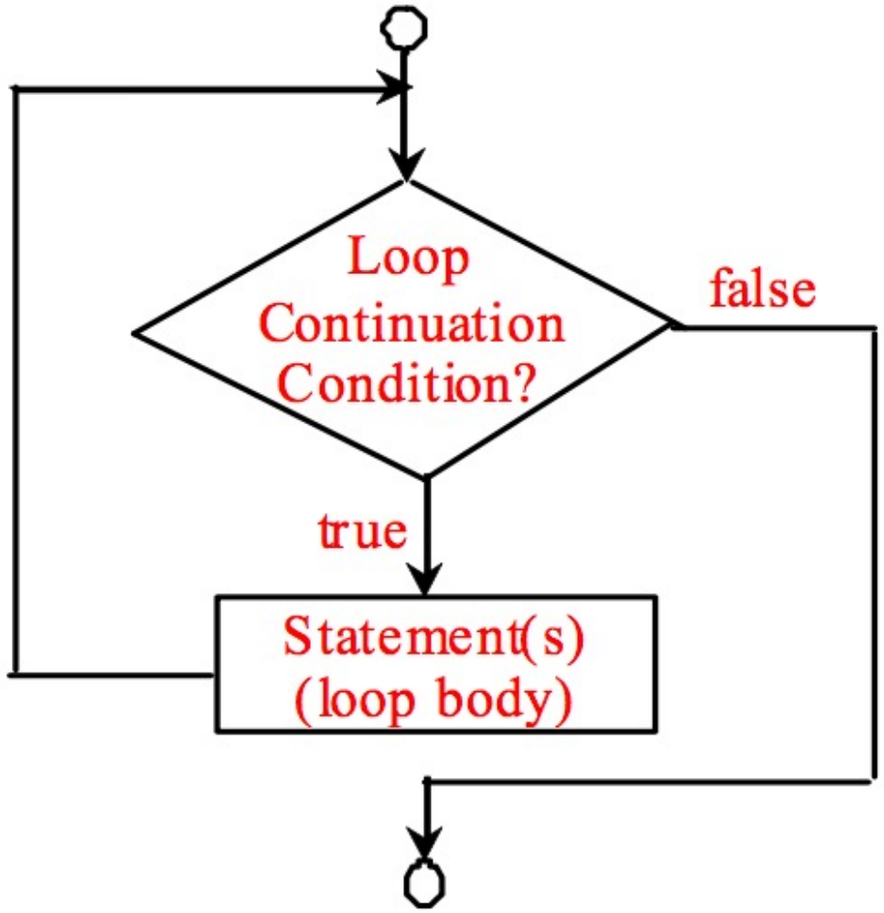
http://introcs.cs.princeton.edu/java/11cheatsheet/

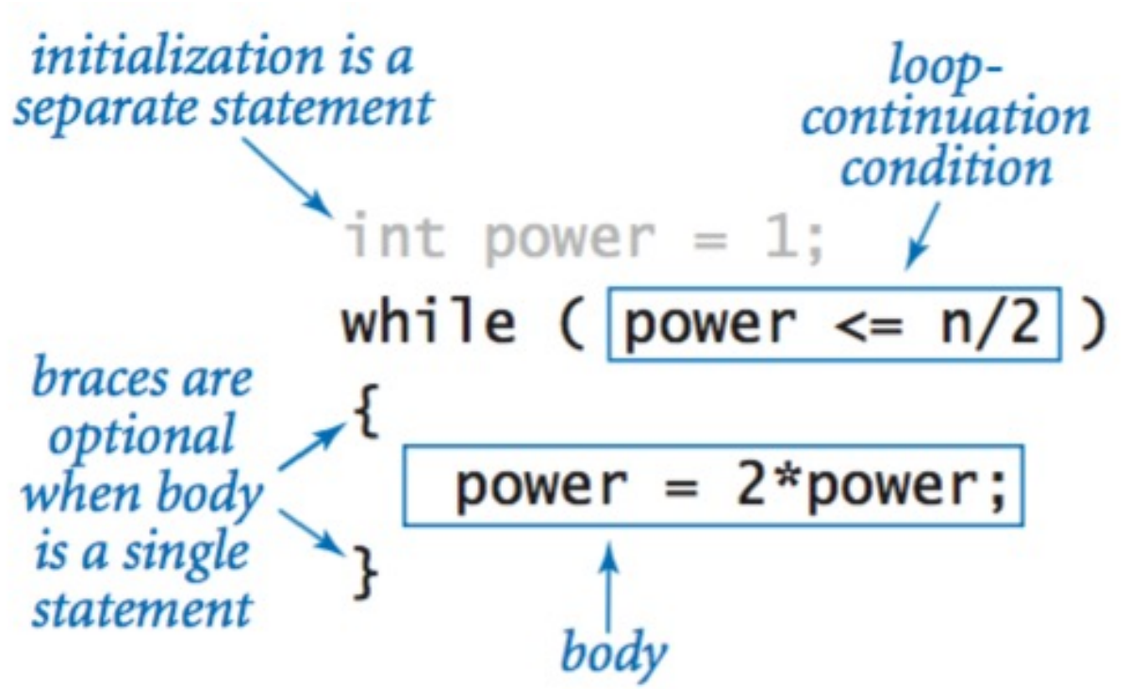
CS170 Java Review Material + Textbook available on Canvas



CS171 [Spring 2023]

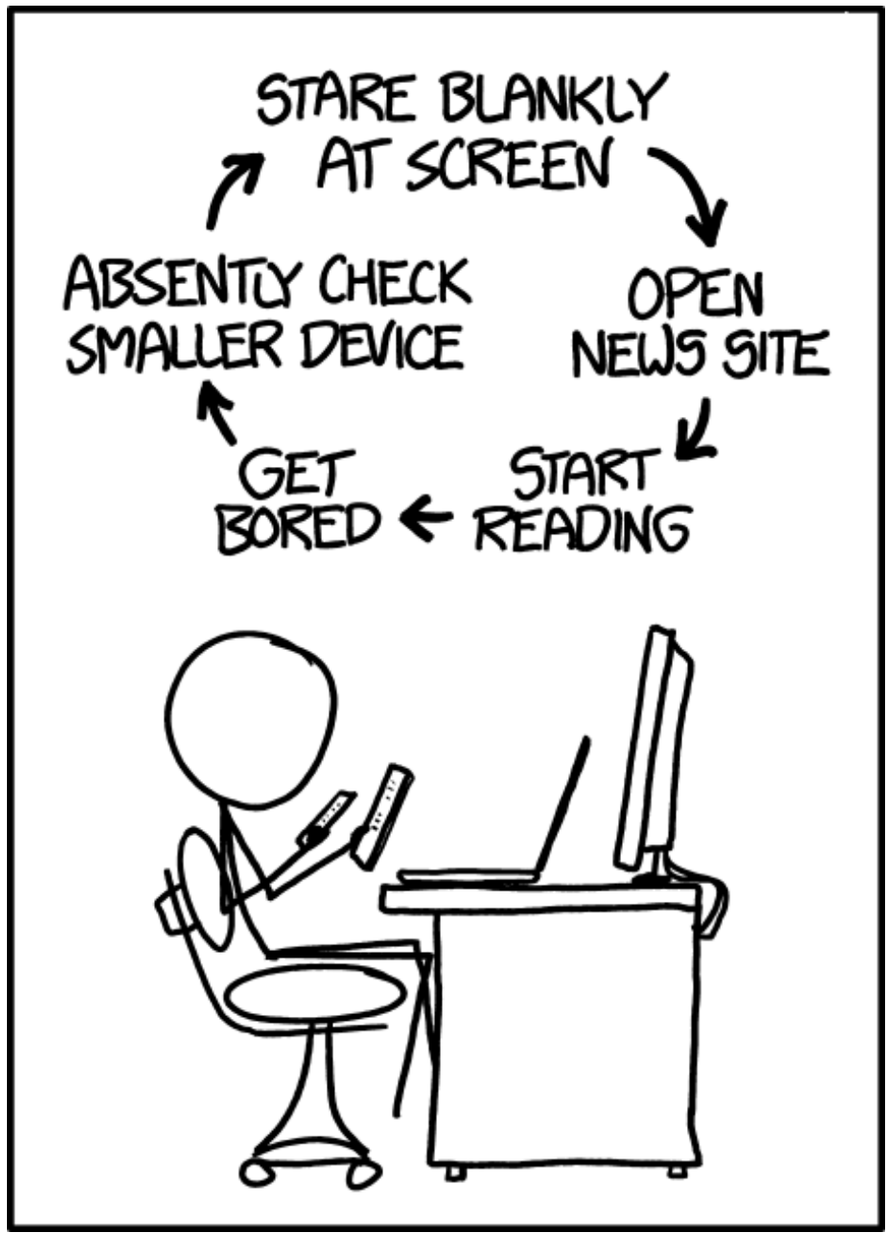
Java Review: While Loop





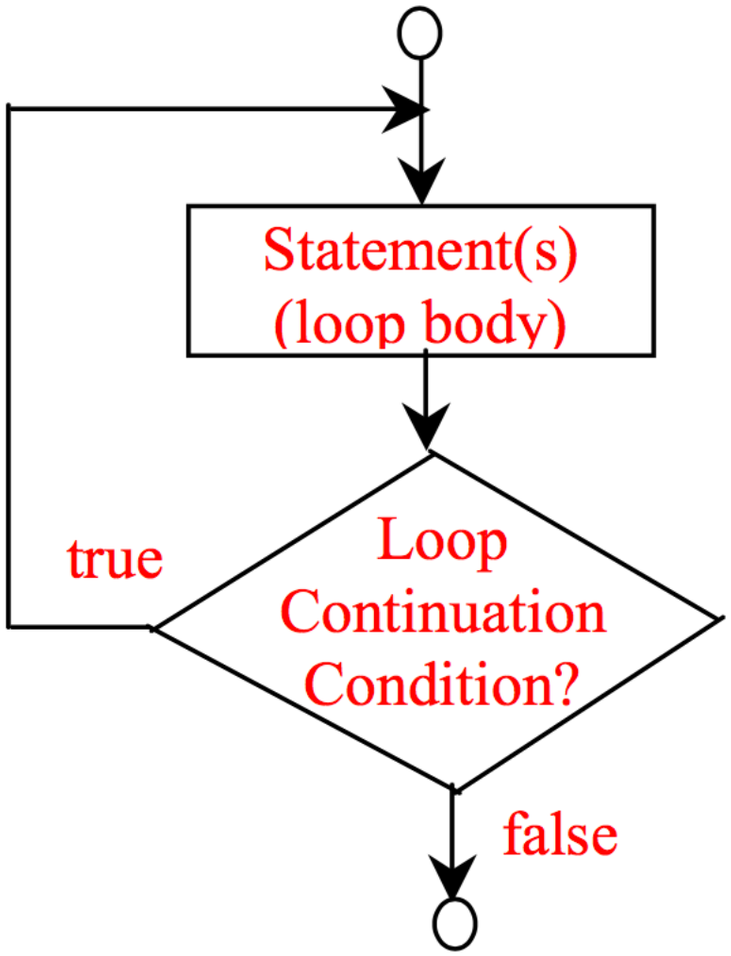
http://introcs.cs.princeton.edu/java/11cheatsheet/

Java Review: Infinite Loops

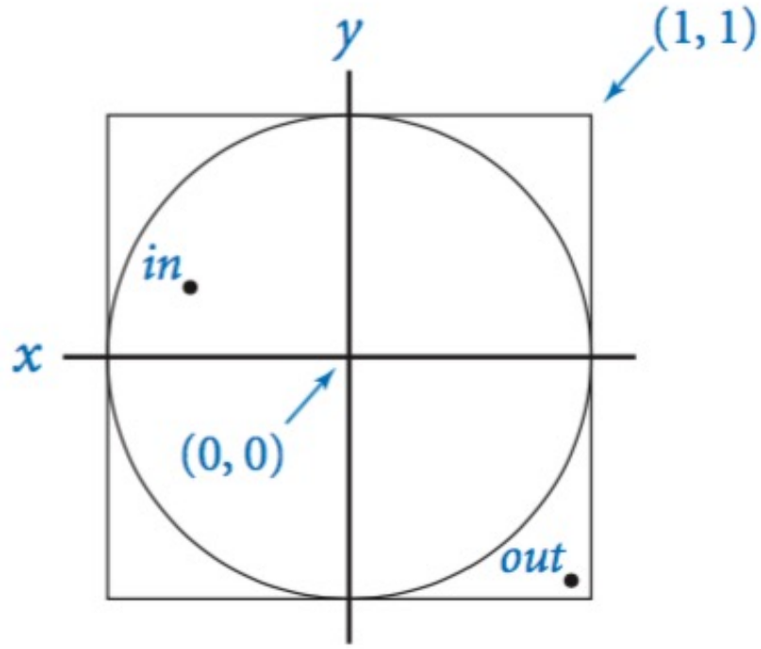


https://imgs.xkcd.com/comics/loop.png

Java Review: Do-While Loop

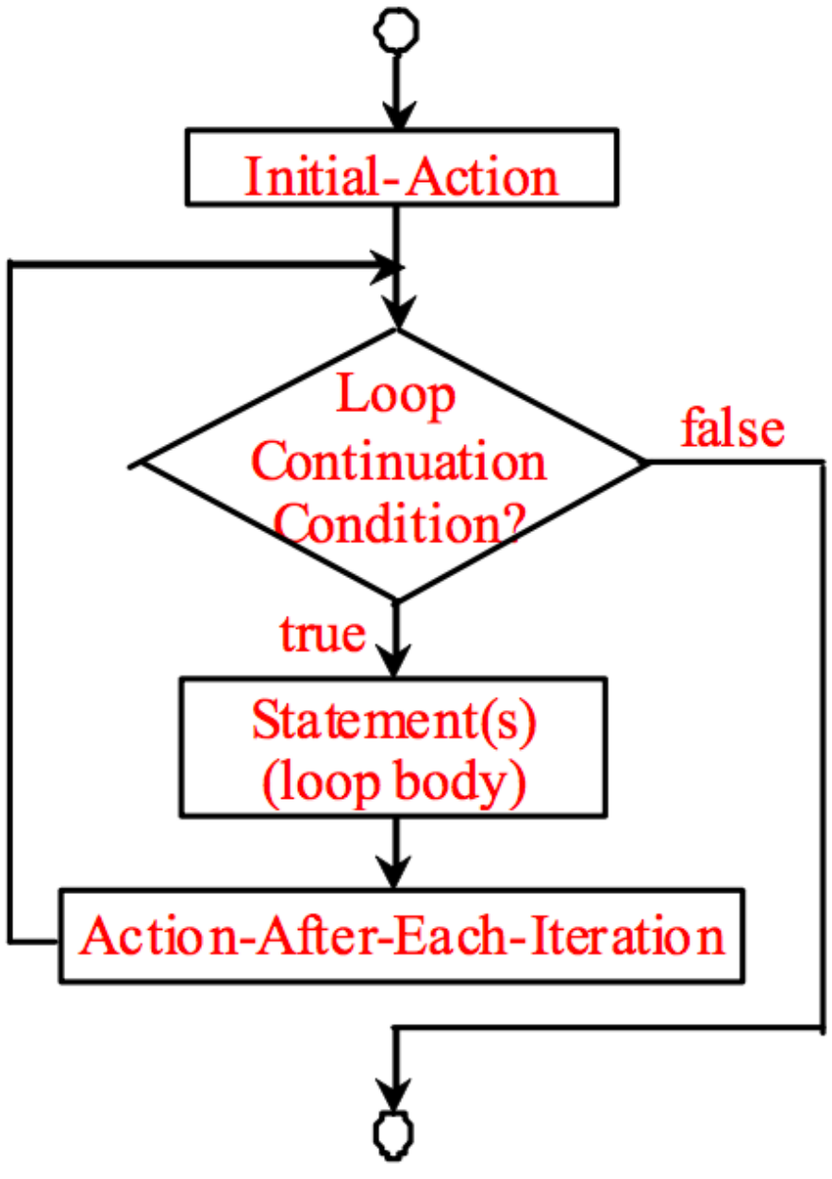


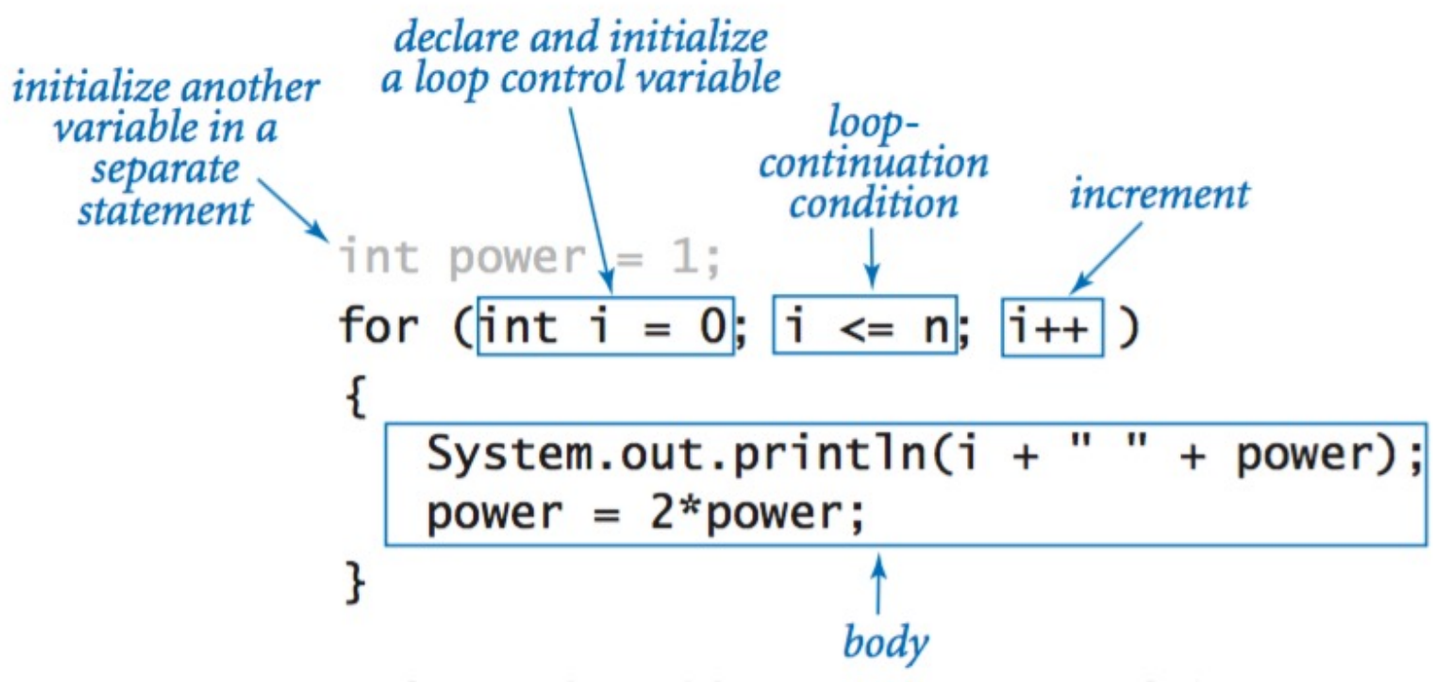

```
do
{  // Scale x and y to be random in (-1, 1).
   x = 2.0*Math.random() - 1.0;
   y = 2.0*Math.random() - 1.0;
} while (Math.sqrt(x*x + y*y) > 1.0);
```



http://introcs.cs.princeton.edu/java/11cheatsheet/

Java Review: For Loop





http://introcs.cs.princeton.edu/java/11cheatsheet/

Java Review: Which Loop?

- Follow your intuition and comfort level
- General rules of thumbs:
  - `for` loop: Known # of repetitions
  - `while` loop: Unknown # of repetitions
  - `do-while`: Loop body should be executed *before* condition check

Java Review: Pretest (Self-Test) Q1

- What is the output of the following code fragment?

```
int sum = 1;
for (int i = 0; i <= 5; sum = sum + i++);
System.out.print(sum);
```

- Output:
  16

What if…

- What is the output of the following code fragment?

```
int sum = 1;
for (int i = 0; i <= 5; sum = sum + ++i);
System.out.print(sum);
```

- Expected output?

Tracing Tip

- For each iteration, record the values of your variables; keep doing that until the loop condition is false

```
Iteration  |  sum  |  i
=================================
1
2
3
4
...
```

Then verify with code!

```java
public class Pretest{
  public static void main(String[] args){
    int sum = 1;
    for (int i = 0; i <= 5; sum = sum + (++i) );
    System.out.println(sum);

  }
}
```

Java Review: Pretest Question 2

- If `a` and `b` are ints such that `b != 0`, then which of the following expressions is always equivalent to `a % b`

A. `a - (a / b) * b;`

B. `(a / b) * b;`

C. `a - a / b;`

D. `(double) a / b - a / b;`

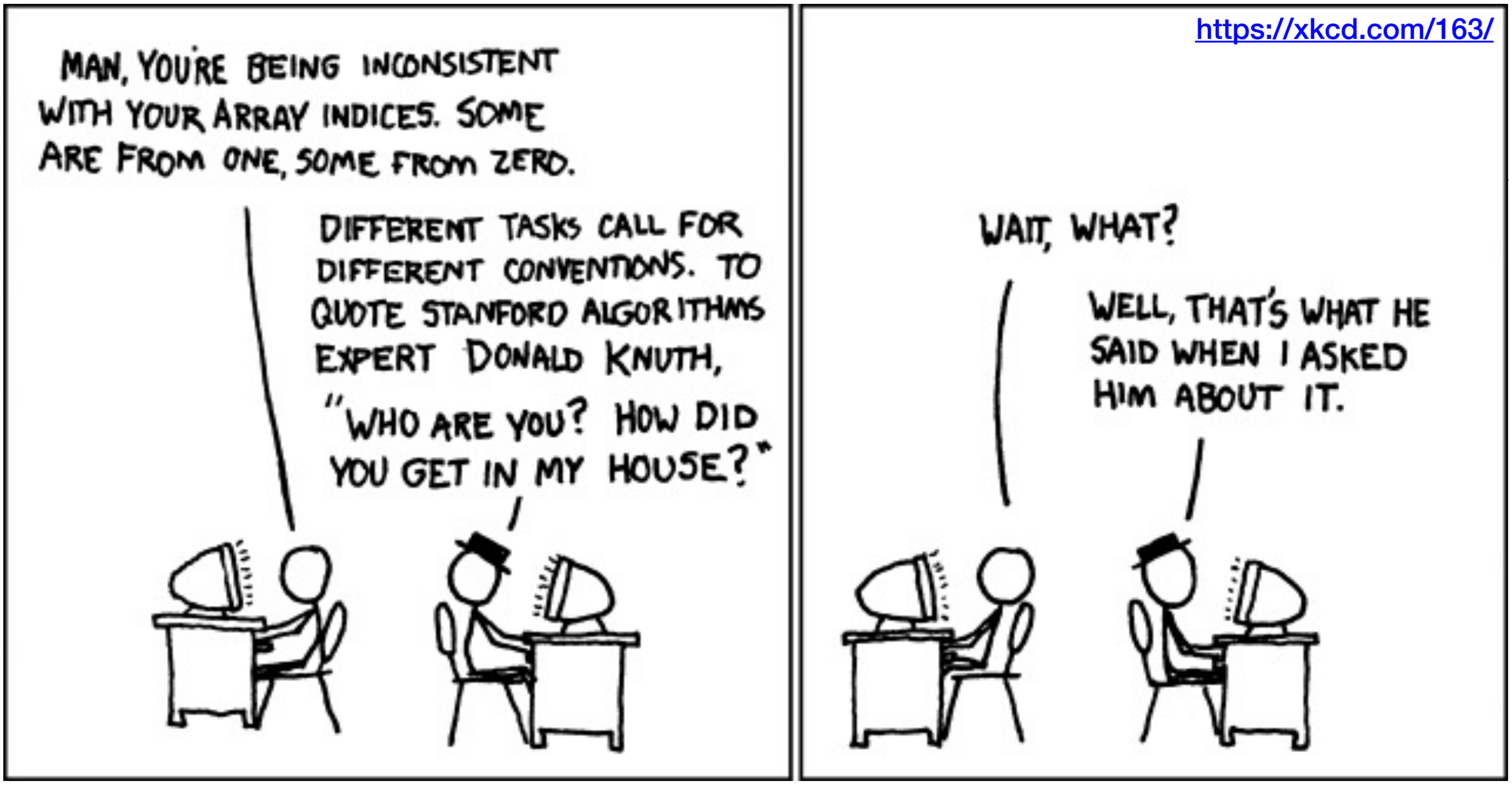
Java Review: Statements

statement	examples	definition
declaration	`int i;` `double c;`	create a variable of a specified type, named with a given identifier
assignment	`a = b + 3;` `discriminant = b*b - 4.0*c;`	assign a data-type value to a variable
initializing declaration	`int i = 1;` `double c = 3.141592625;`	declaration that also assigns an initial value
implicit assignment	`i++;` `i += 1;`	`i = i + 1;`
conditional (`if`)	`if (x < 0) x = -x;`	execute a statement, depending on boolean expression
conditional (`if-else`)	`if (x > y) max = x;` `else max = y;`	execute one or the other statement, depending on boolean expression

loop (`while`)	`int v = 0;`
	`while (v <= N)`
	`  v = 2*v;`
	`double t = c;`
	`while (Math.abs(t - c/t) > 1e-15*t)`
	`  t = (c/t + t) / 2.0;`
		execute statement until boolean expression is `false`

loop (`for`)	`for (int i = 1; i <= N; i++)`
	`  sum += 1.0/i;`
	`for (int i = 0; i <= N; i++)`
	`  StdOut.println(2*Math.PI*i/N);`
		compact version of `while` statement

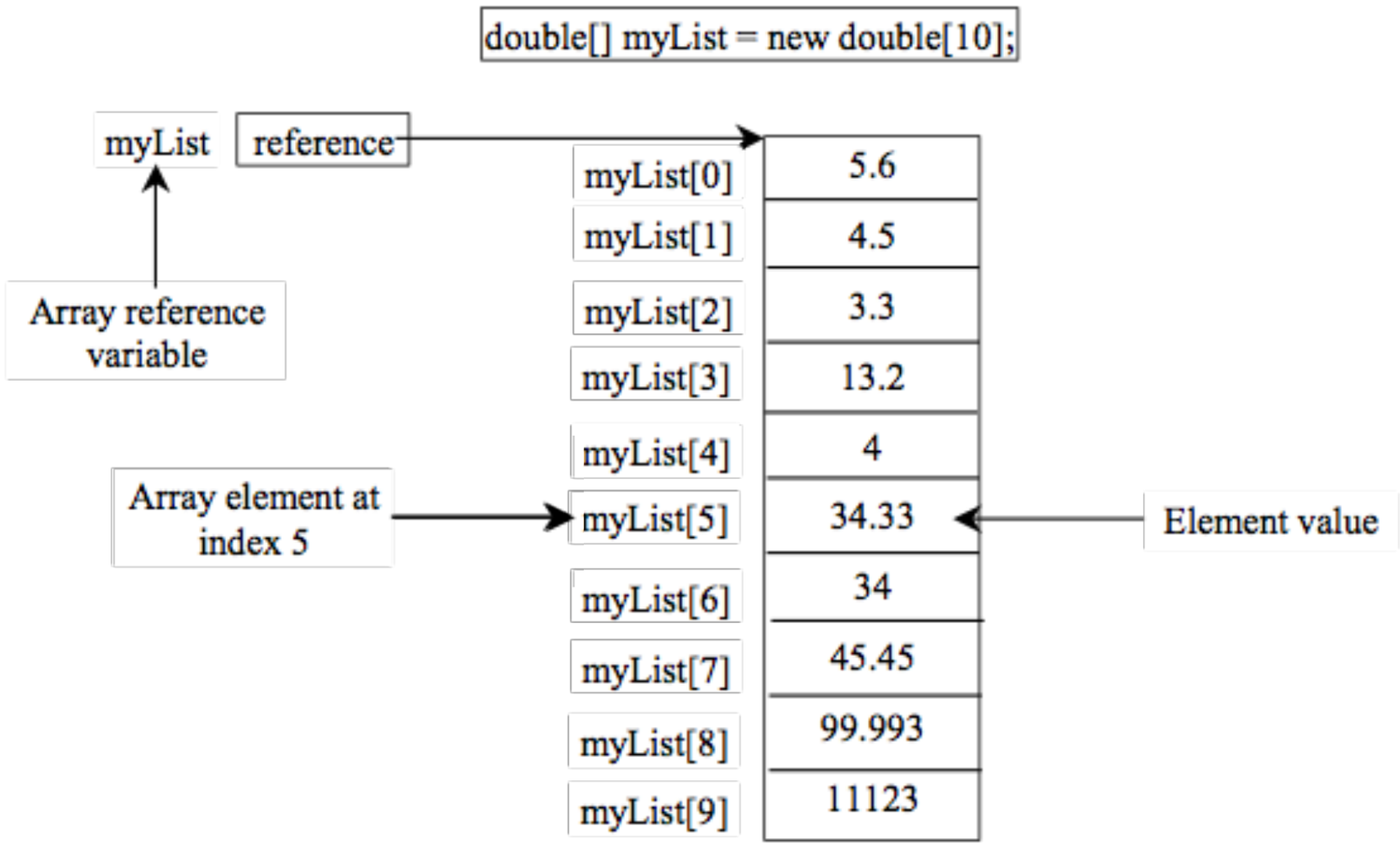
call	`int key = StdIn.readInt();`	invoke other methods (see page 22)
return	`return false;`	return from a method (see page 24)



https://xkcd.com/163/

Java Review: Arrays

- Array is a data structure that stores a sequence of values that are of the same type



Java Array: Initialization

- For primitive data types, all elements are initialized to be **zero** by default

```
int[] myarray = {2,3,5,7,9,11,13,17};
int[] foo = new int[42];
```

- For object data types…?

- For object data types, all elements are initialized to **null** by default

```
Circle[] cirArray = new Circle[100];
```

Java Array: Length

- Once created, the size is fixed — cannot be changed
- Find the size by using the property `length`

```
arrayrefVar.length;
```

- Example:

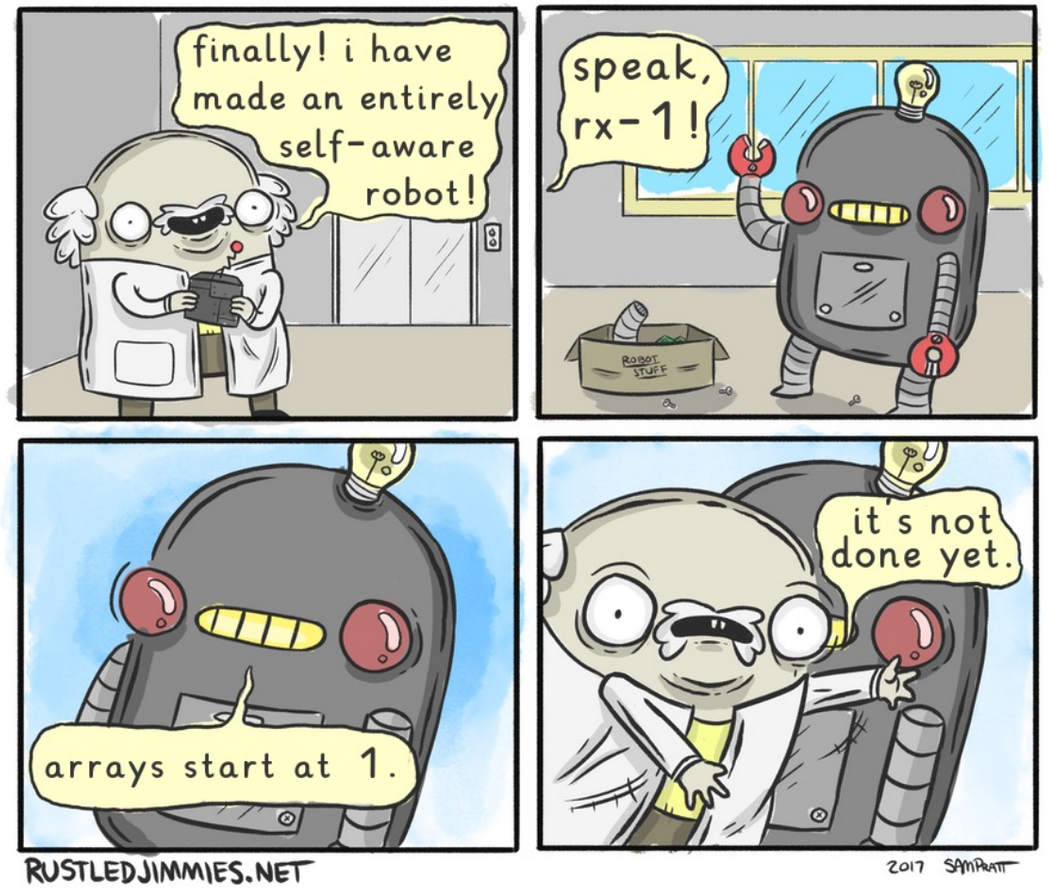
```
int[] numbers = new int[10];
int len = numbers.length; // 10
```

Arrays: Accessing Elements

- Elements are accessed through the index
- Each element in the array is represented using

```
arrayRefVar[index]
```

- Java uses zero-based indexing: starts at 0 and ends in array.length - 1



Processing Arrays

- Use for loops to traverse or process array elements
  - Elements are of the same type ✓
  - Size of the array is known (*arrName.length*) ✓

```
for (int i = 0; i < myList.length; i++) {
    // process ith element myList[i]
    System.out.println(myList[i]);
}
```

Typical Array Processing

- Find maximum / minimum of array values
- Compute the average / sum of array values
- Copy to another array
- Reverse the elements within an array

Copying Array

```
double[] a = {2,3,5,7,9,11,13,17};
double[] b = a;

a[0] = 200;
System.out.println(b[0]);
```

What gets printed?

200.0

Let's code this… [Pretest Q3]

Q: Write a method named inorder that takes an array of integers as a parameter and returns the count of elements that are *smaller* than the element immediately following it.

> For example, if the array consists of 3, 7, 8, 5, 4, 9, the method should return: 3 because 3 < 7, 7 < 8, and 4 < 9

```
for (int i = 0; i < myList.length; i++) {
    // process ith element myList[i]
    System.out.println(myList[i]);
}
```

Java Review: Primitive Data Types

- Integers with arithmetic operations:
  `byte`, `short`, `int`, `long`

- Real numbers with arithmetic operations:
  `float`, `double`

```
double pi = 3.14;
```

- Boolean operations with {`true`, `false`} values and logical operations: `boolean`

- Characters: `char`

Java Review: Object Data Types

- Pretty much everything else!
- `array` types
- Special `null` types
- Class types
  - Example of a class type we used a lot in CS170: `String`
- Interface types

What if we need more sophisticated data types?

- Example:
  A bank manager needs a program to store and update credit card information for customers

- Your job is to store this info for all issued credit cards:
  - Card Type (“Silver” or “Gold”)
  - Customer ID
  - Card Number (14-digit number)
  - Card Expiration Month (e.g. “03”)
  - Card Expiration Year (e.g. “2021”)
  - Card Limit