CS170: Introduction to Computer Science I
Sections 1, 2, 9 (2:30-3:45 PM)

Today: Introductions, Overview, and Logistics

Introduction

- My name is : *Seyed* **Navid Hashemi** *Tonekaboni*
- You can call me **Navid**
- Email: navid.ht@emory.edu
- Office: MSC W302-J.

I am a Visiting Assistant Professor at the Department of Computer Science.

What is this course about?

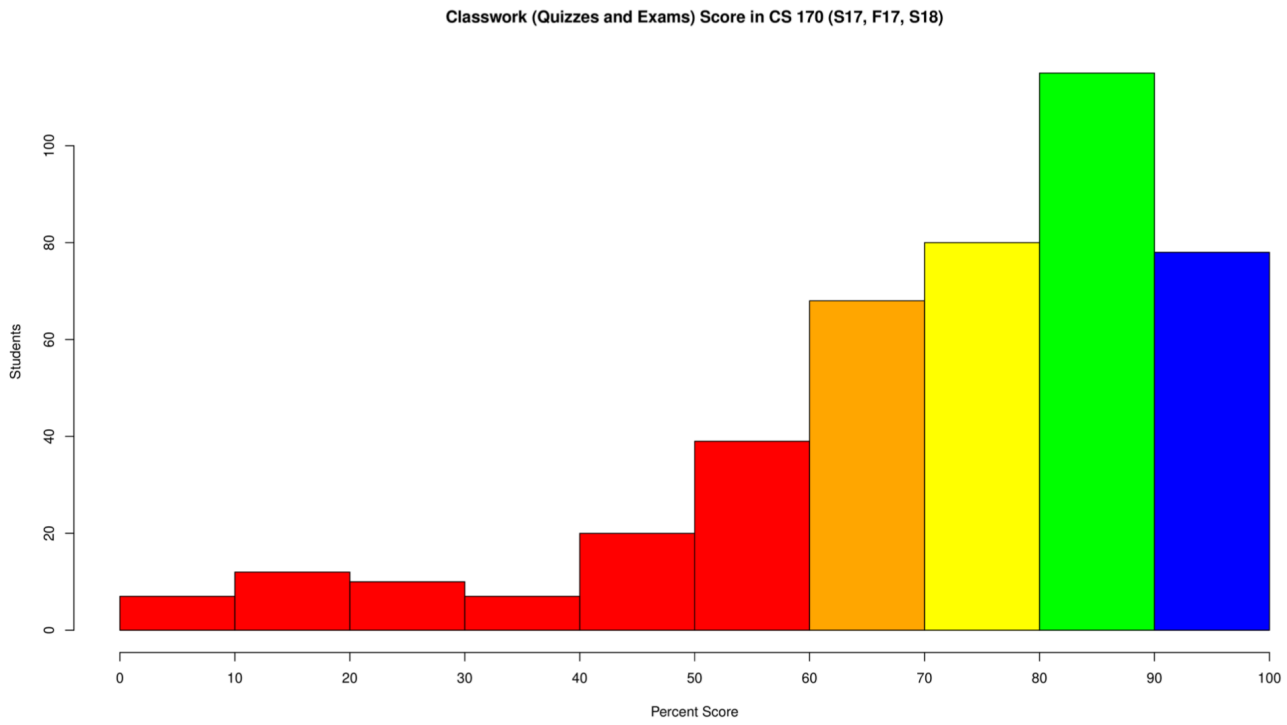
- You will learn…
  - Programming concepts and principles using Java programming language

  - How to think like a Computer Scientist; problem solving and programming

No previous CS or programming knowledge expected

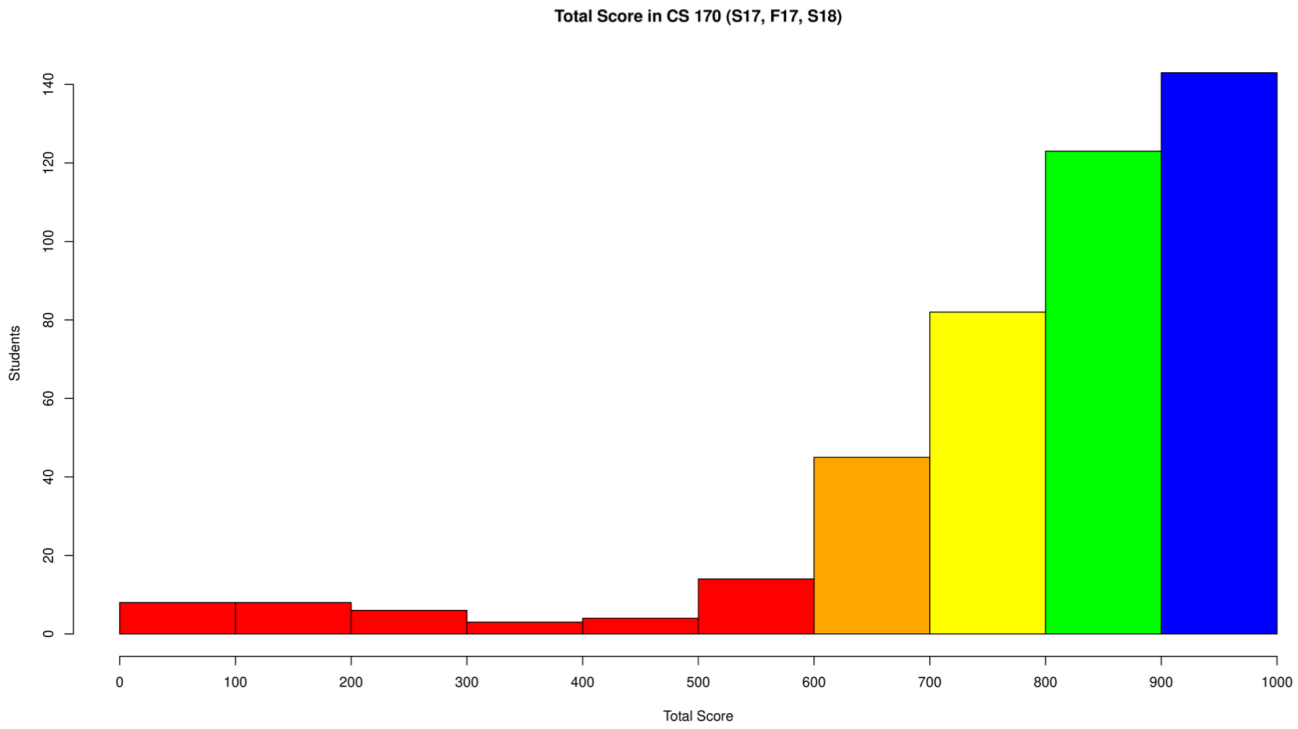
Myths and Facts about CS170

- **Myth:** CS 170 is impossible to pass!
- **Fact:** Most students do very well in CS 170



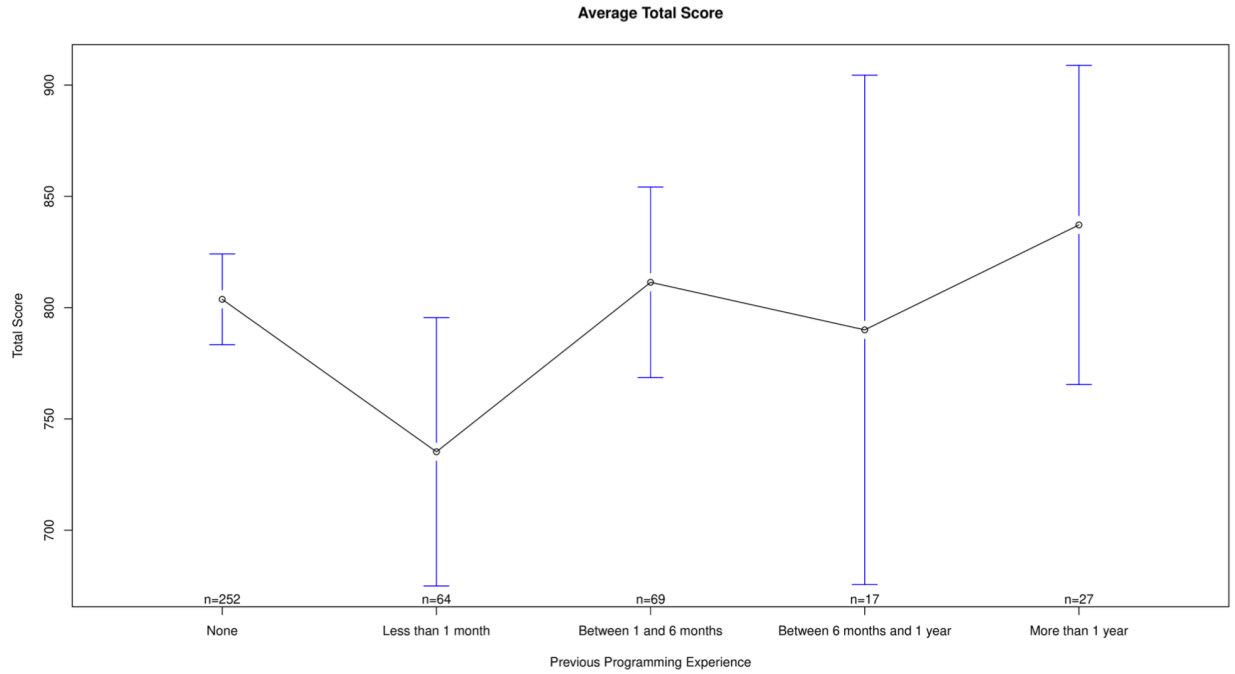
Myths and Facts about CS170

- **Myth:** CS 170 is impossible to pass!
- **Fact:** Most students do very well in CS 170



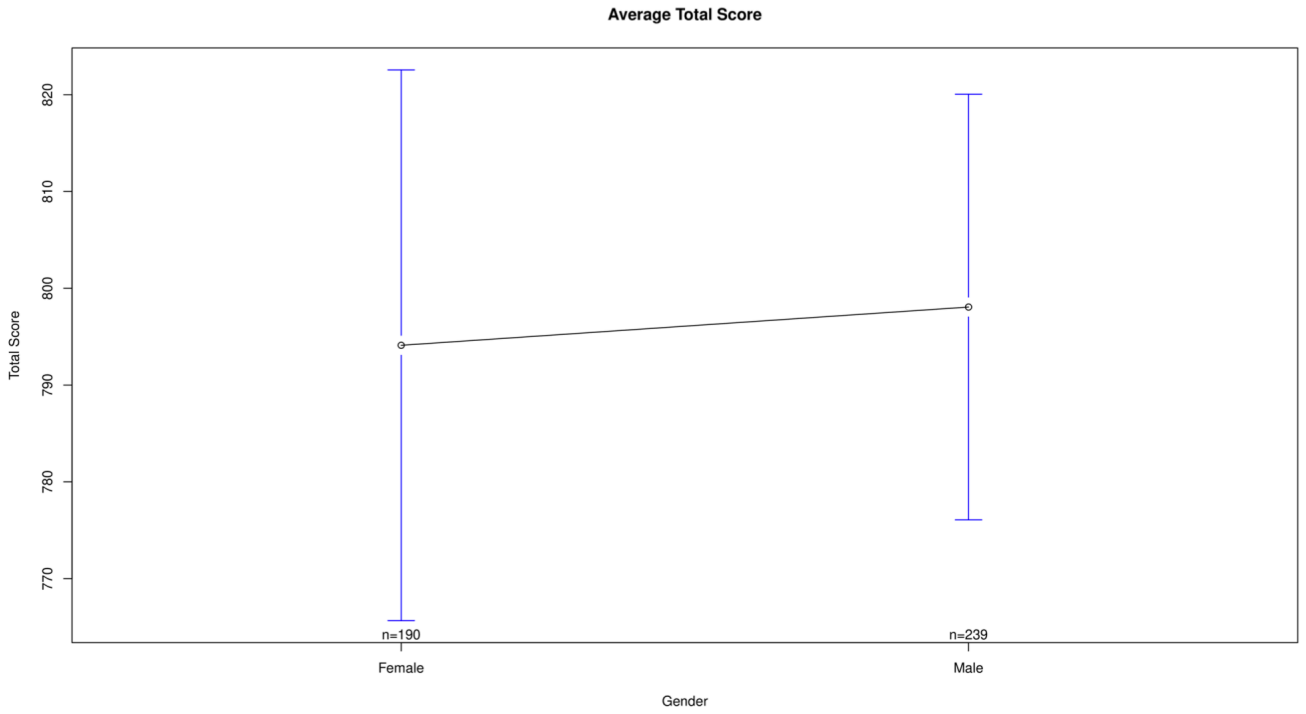
Myths and Facts about CS170

- **Myth:** Students with previous programming experience have an unfair advantage in CS 170
- **Fact:** Students with no previous programming experience(majority of the class) perform as well as those with prior experience in CS 170



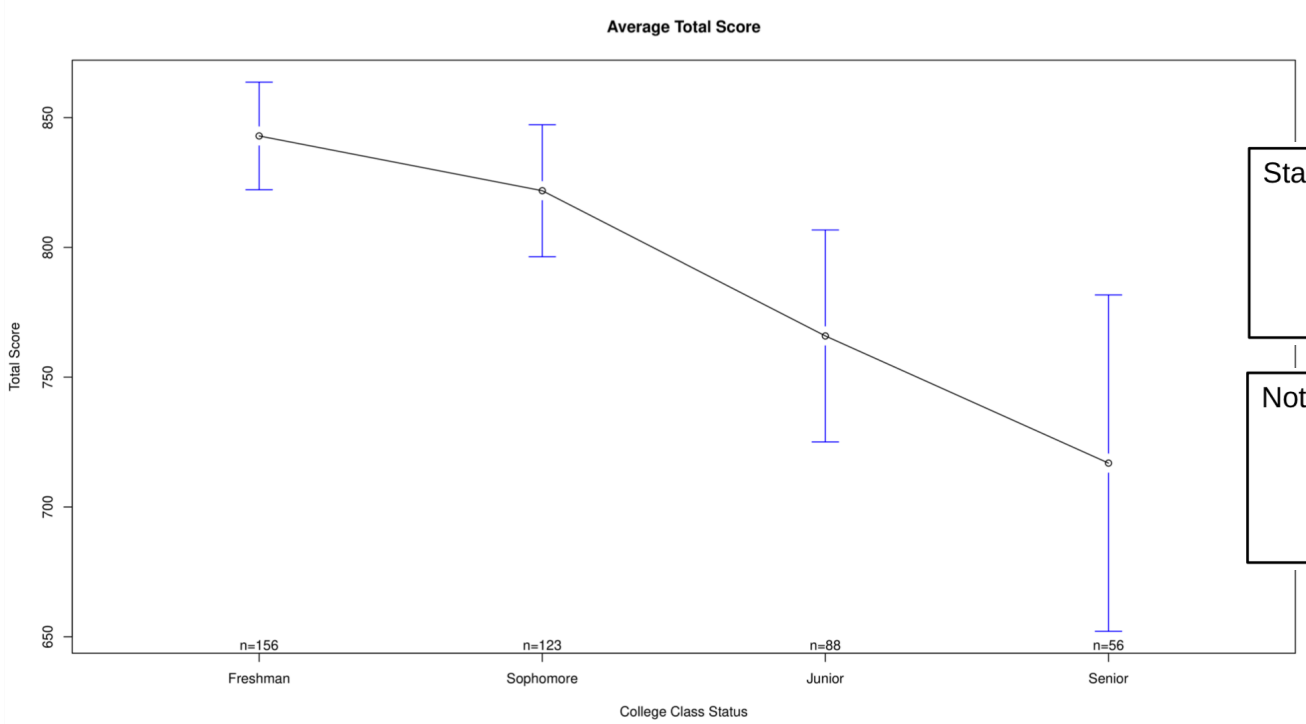
Myths and Facts about CS170

- **Myth:** Women are not good at Computer Science
- **Fact:** Men and women perform equally well in CS 170



Myths and Facts about CS170

- **Myth:** I am a junior/senior, so I can easily breeze through a 100-level course such as CS 170
- **Fact:** Juniors and seniors tend to perform worse than freshmen and sophomores in CS 170

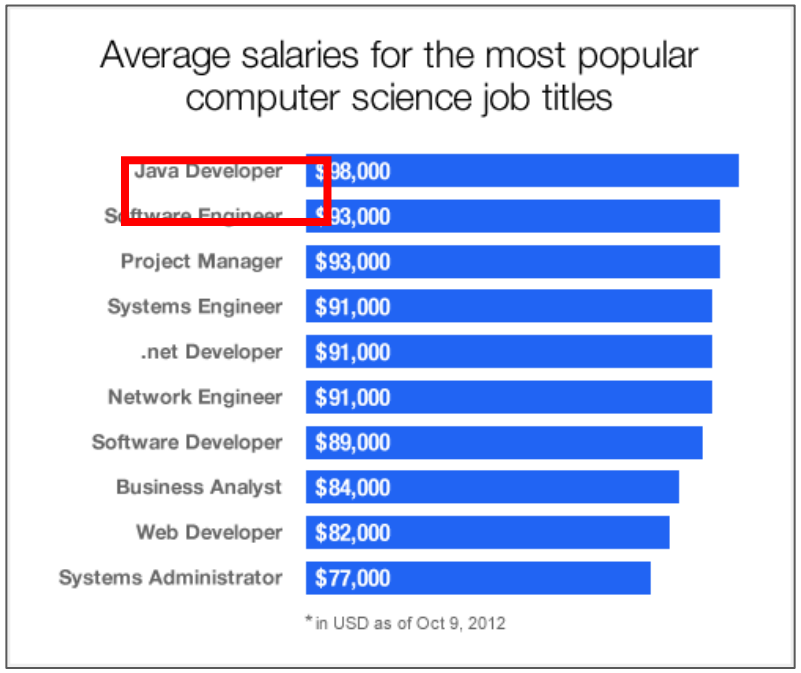


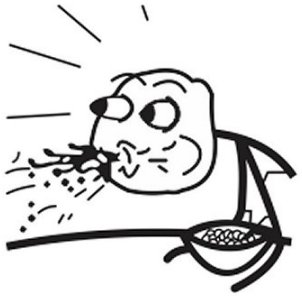
Statistically significant differences:
freshmen > juniors
freshmen > seniors
sophomores > juniors
sophomores > seniors

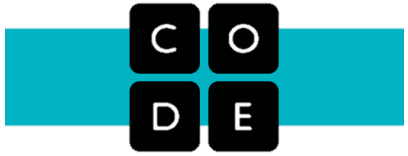
Not statistically significant differences:
freshmen ~= sophomores
juniors ~= seniors

It's a smart choice to take CS170…

Hardware is Cheap, Programmers are Expensive

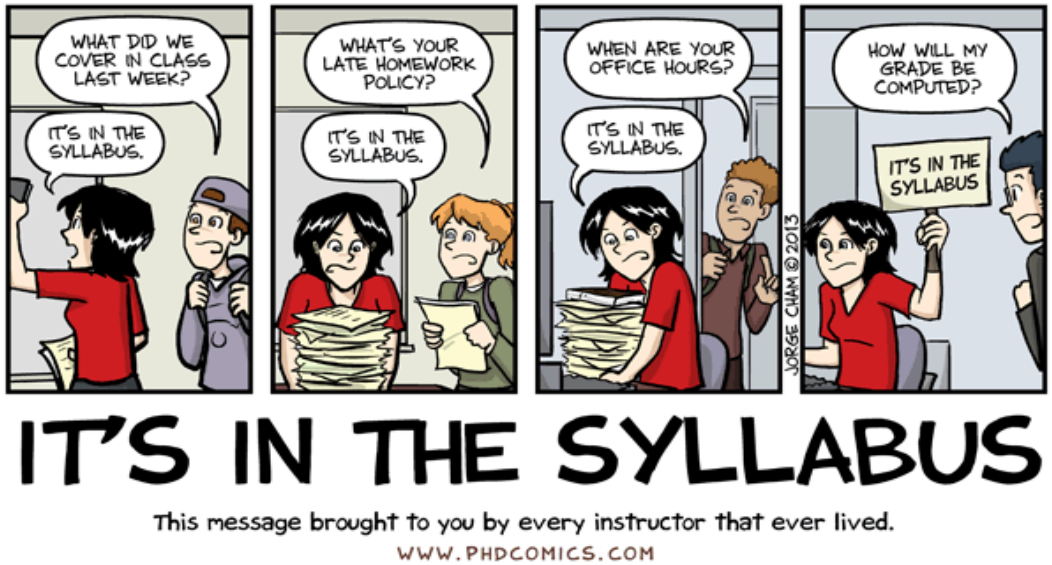






https://youtu.be/nKlu9yen5nc

Course logistics…



CS170 Learning Activities

- Lectures. Mix of new material presentation and interactive problem solving.
  ➔ Consistent attendance crucial for grasping the material & performing well on quizzes, assignments, exam!

- Lab Quizzes (Fridays).

- Homework Assignments.

Reinforce your understanding of programming concepts introduced in lectures

- Peer Evaluations.

Strengthen your code interpreation & debugging skills

CS170 Communication Platform

- We will be using Piazza for class discussion and as a repository for all class resources (lecture material, etc.)

- Your questions about course material should be posted to Piazza, not emailed to teaching staff
  - Use email for private inquries or appointment requests

You are expected to stay on top of everything posted on Piazza at all times

Don't post any code!

piazza

Teaching Staff

- Instructor (Sections 1-2-9): Dr. Navid Hashemi
  - Email: navid.ht@emory.edu
  - Office: MSC W302-J. Office Hours: Wed 4:30-5:30PM; Thu 10-11AM
  - Email me if you need to schedule an appointment

- Co-Instructor (Sections 3 to 8)
  - Dr. Davide Fossati
- Piazza page covers *all* sections (Benefit from everyone's questions & discussions)

- TAs: will be holding Office Hours and Labs.
  Details will be available next week under Piazza ➔ "Staff" tab

CS170 Grading

Item	Count	Points	Total
Quizzes	12	50	600
Final exam	1	100	100
Homework	5	40	200
Peer evaluations	100-150		100
Total			**1000**

Letter grade	Minimum points
A	933
A-	900
B+	866
B	833
B-	800
C+	766
C	733
C-	700
D+	666
D	600
F	0

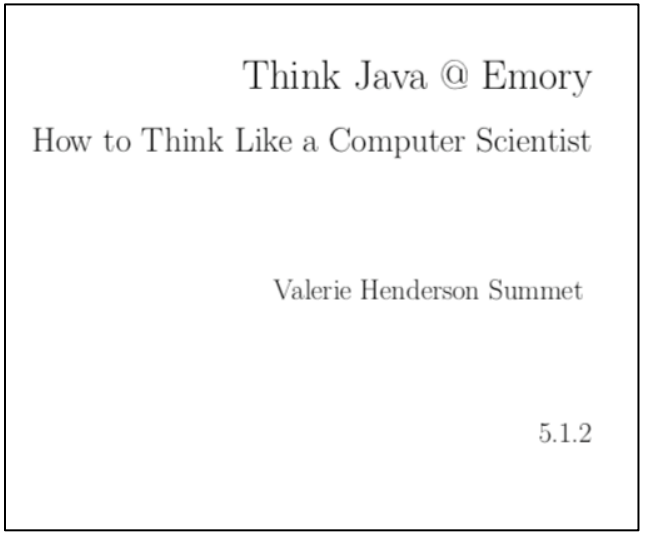
Quizzes Policy

- 12 Quizzes in total (on Fridays; dates available in Syllabus).
- Drop-2 Policy. If you miss up to 2 quizzes, the missing ones will be dropped and replaced with the average of other quizzes & final exam.
- Drop-2 Policy is not a bonus, it's designed to account for all circumstances such as sickness, family emergencies, sports, religious holidays, etc.
- No need to provide justification for missing up to 2 quizzes.
- No make-up quizzes will be given for such circumstances.
- For extreme circumstances (causing you to miss more than 2 consecutive quizzes) ➔ contact me ASAP.

See Syllabus (uploaded on Piazza) for details

Textbook

- "Think Java @ Emory" by Valerie Henderson Summet
  - Available on Piazza => "Resources"
    http://www.mathcs.emory.edu/~valerie/textbook/thinkjavaemory.pdf
- The textbook covers most of the material presented in the course, but not necessarily in the same order.
  - Important for understanding many lecture topics
  - Helpful code examples, glossary summaries, exercises, etc.
  - Flow of concepts is smooth and easy to follow :-)


Think Java @ Emory

How to Think Like a Computer Scientist

Valerie Henderson Summet

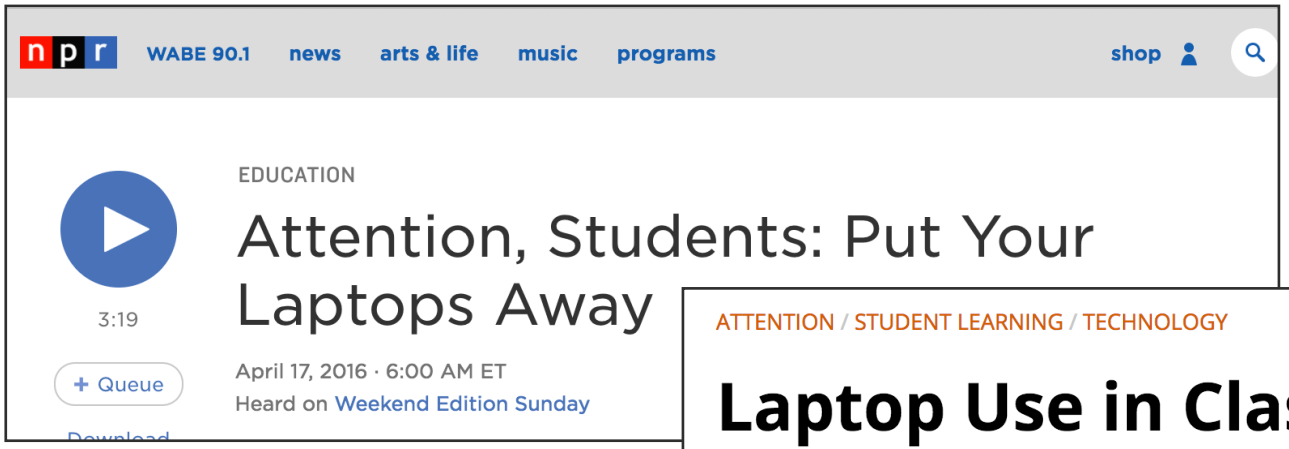
5.1.2

Bonuses and Penalties

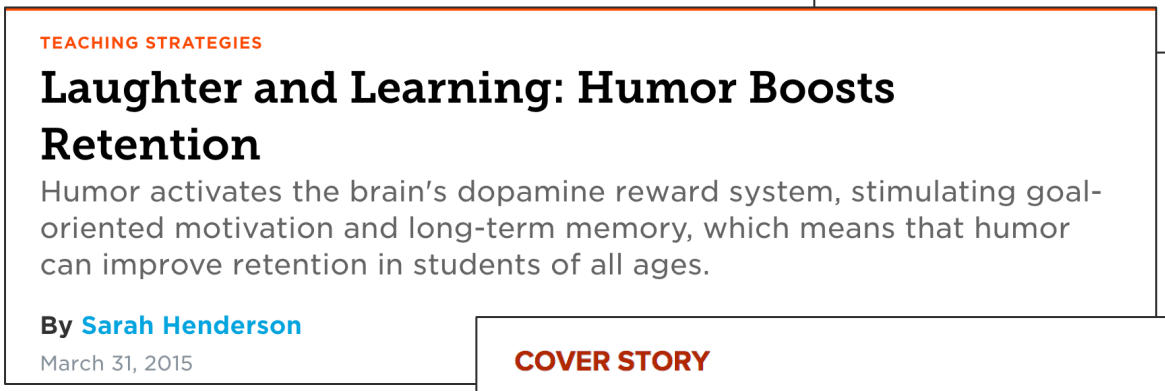
- Homework early submission bonus. If you submit an entire homework assignment no later than 48 hours before the deadline, and the total score on the rest of the homework assignment is at least 20 points, you will receive 2 bonus points.
- Typo catcher bonus. If you discover a typo or other mistake in a homework assignment, quiz, or exam, and you are the first person to report it, you will receive 1 bonus point.
- Piazza participation award. The top three students writing the largest number of endorsed answers on Piazza will receive 5 bonus points.
- Inappropriate regrade request penalty. If you submit a regrade request which results in no score change, you will receive no penalty for the first occurrence, and -1 point penalty for each subsequent inappropriate regrade request occurrence.

See Syllabus (uploaded on Piazza) for details

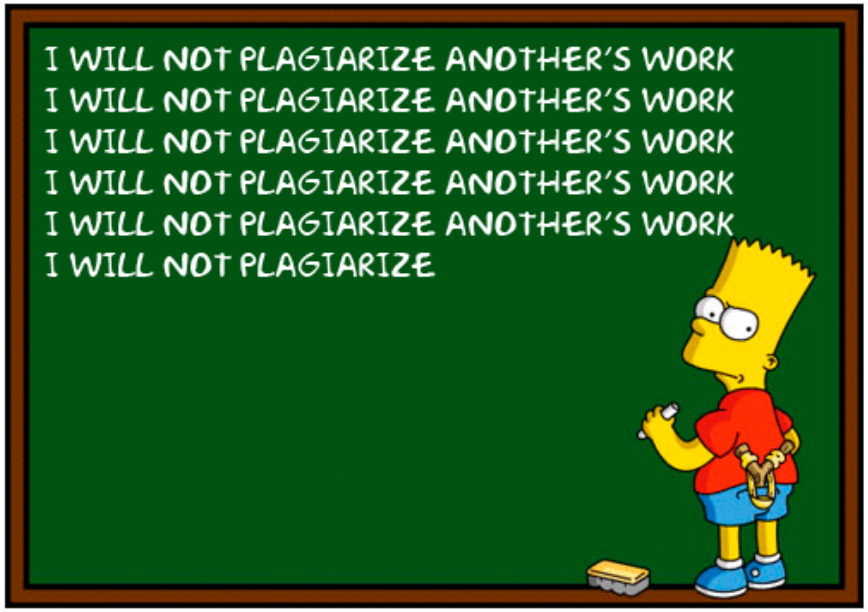
Electronic Devices (when we're not coding…)



On the other hand…



Academic Plagiarism & Dishonesty



http://world.edu/academic-plagiarism

Honor Code

- All students are expected to follow Emory's Honor Code, particularly Article 4: Academic Misconduct.
  http://catalog.college.emory.edu/academic/policies-regulations/honor-code.html

- The Computer Science Dept. has a specific policy regarding the submission of computer code. http://www.mathcs.emory.edu/spca.php

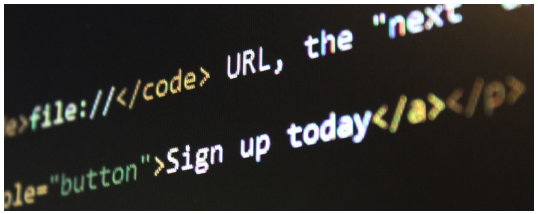
- Most frequent violations in CS170 are cheating and excessive collaboration.
  ➔ See examples in Syllabus

- Goldern rule: Never show your code to other people, except your teammates, or TAs/course instructors (when necessary).

See Syllabus (uploaded on Piazza) for details

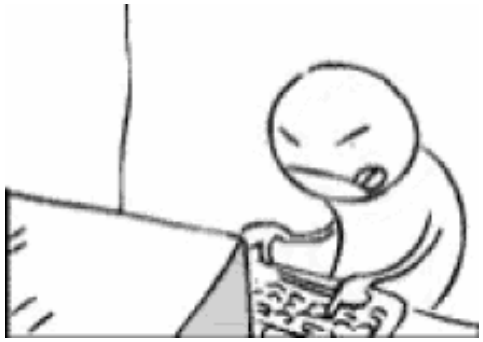
Now back to course content…

“The way of the program”

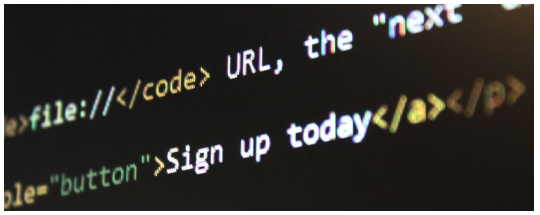


- Most important skill for a computer scientist is problem-solving
  - Learning how to program is great practice for problem-solving!

- So what is programming?



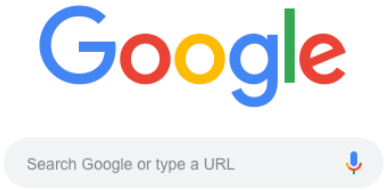
"The way of the program"

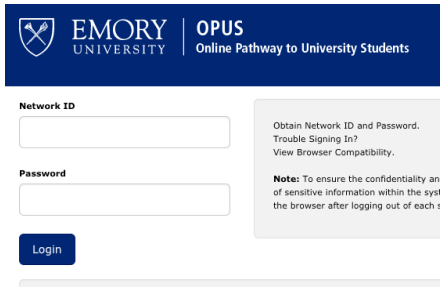


- Most important skill for a computer scientist is problem-solving
  - Learning how to program is great practice for problem-solving!

- So what is programming?
  - Sequence of instructions (statements) that specify how to perform a certain computation.
  - Programming is involved in so many aspects of our lives today.

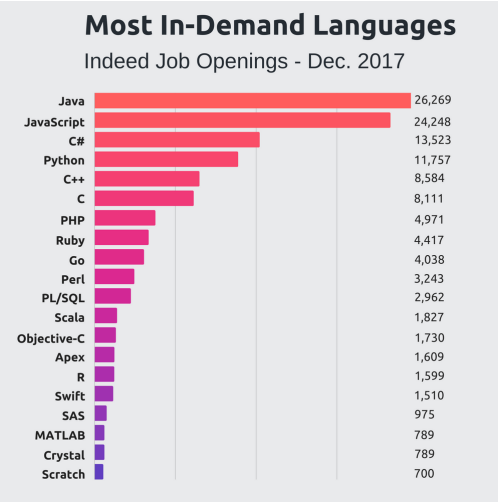
We will learn lots of new terminology in CS170; you will soon get used to it ;-)





Java Programming Language

- Java is a popular, widely used programming language



- Java enables developing applications for the desktop computers, smart phones & tablets, Internet websites, web servers

- Java is a high-level programming language

Levels of Programming Languages

Executable Machine code

```
0001001001000101
0010010011101100
10101101001...
```

Binary Language

https://www.slideserve.com/nami/levels-of-programming-languages

Levels of Programming Languages

High-level program

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

Low-level program

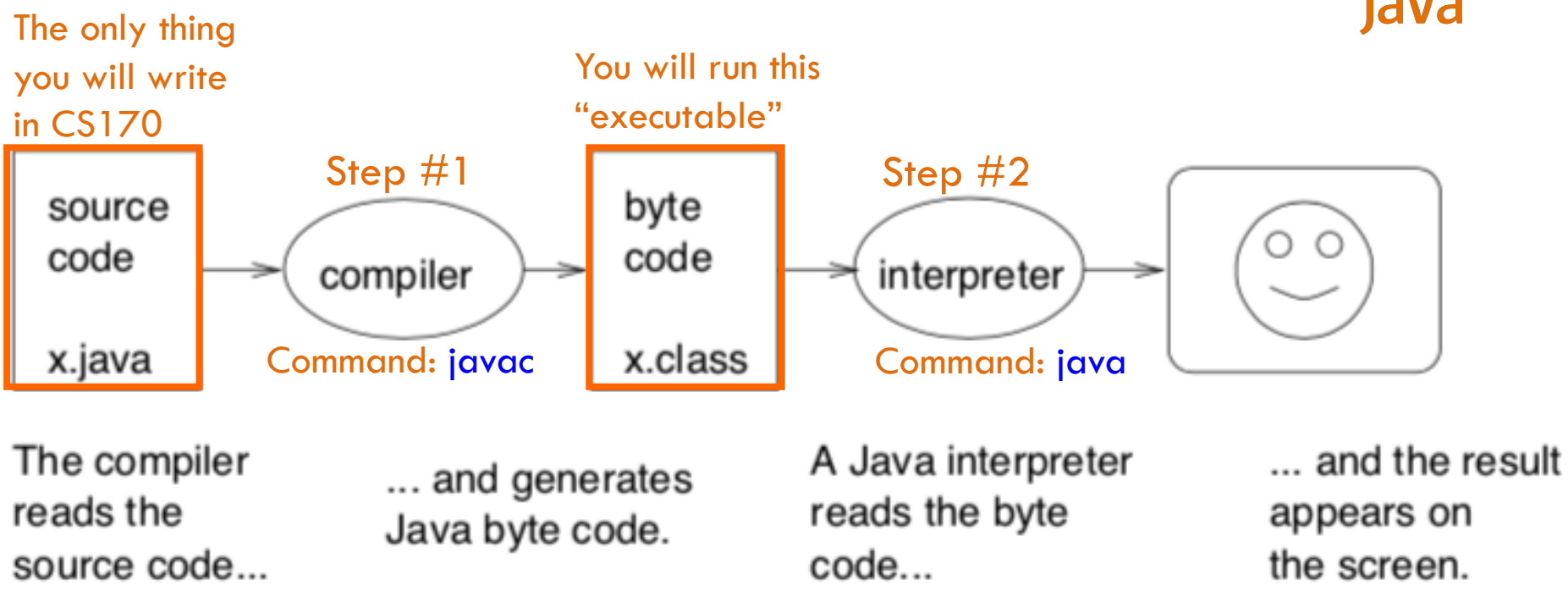
```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

Executable Machine code

```
0001001001000101
0010010011101100
10101101001...
```

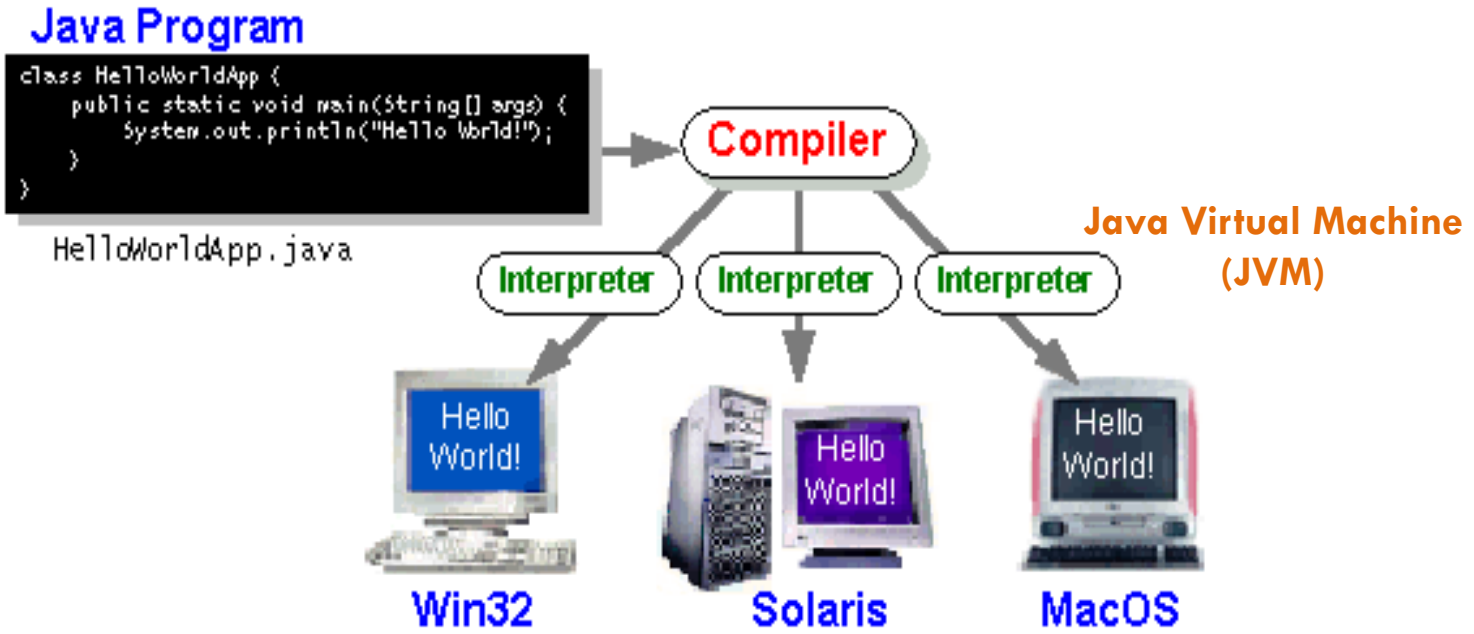
https://www.slideserve.com/nami/levels-of-programming-languages

How do we "run" a Java program?



- We often compile the code once, then run the compiled code later (possibly on different machines too!)

Java Programming Language



https://www.math.uni-hamburg.de/doc/java/tutorial/getStarted/intro/definition.html

Two Java commands we will use a lot

To Compile:

```
javac MyProgram.java
```

To Run:

```
java MyProgram
```

Characteristics of Java

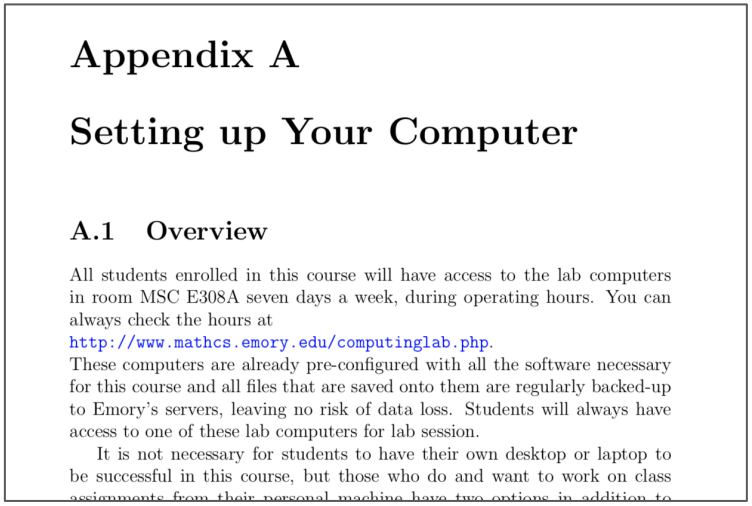
- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java Is Multithreaded
- Java Is Dynamic

We'll talk later about some of these :-)

www.cs.armstrong.edu/liang/JavaCharacteristics.pdf

Setting up your computer

- Lab Computers are set-up and available for CS170 students during operation hours (MSC 3rd floor)
- Setting up your personal laptop



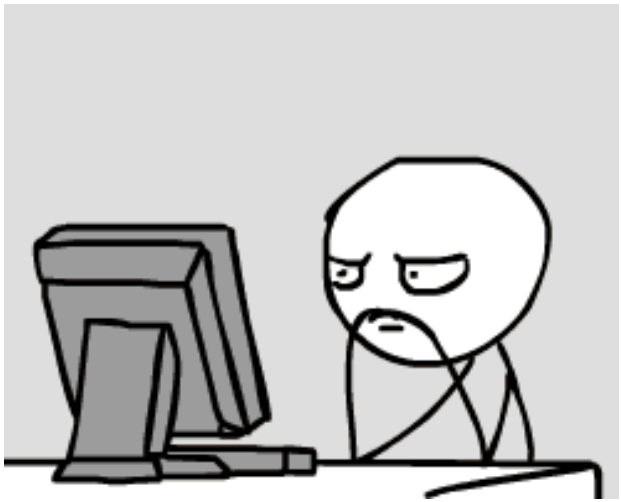
Appendix A

Setting up Your Computer

A.1 Overview

All students enrolled in this course will have access to the lab computers in room MSC E308A seven days a week, during operating hours. You can always check the hours at
http://www.mathcs.emory.edu/computinglab.php.
These computers are already pre-configured with all the software necessary for this course and all files that are saved onto them are regularly backed-up to Emory's servers, leaving no risk of data loss. Students will always have access to one of these lab computers for lab session.

It is not necessary for students to have their own desktop or laptop to be successful in this course, but those who do and want to work on class assignments from their personal machine have two options in addition to



- Additional instructions for Windows10 users will be avialable on Piazza

Some common basic programming operations

- input
  - read data from keyboard, file, etc.
- output
  - display data on screen, file, etc.
- math
  - perform mathematical operations
- testing
  - check for conditions ➔ apply appropriate statements
- repititon
  - perform some action repeatedly (usually with some variation)

Our first program

- Usually the first program prints out a variation of "Hello world!"

```
public class Hello{
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

All of this will make sense, I promise

Java programs are made up of **class definitions**

```
public class CLASSNAME {
  public static void main(String[] args) {
     STATEMENTS
  }
}
```

Class name is chosen by the programmer (you!)

"main" is a **method** (i.e. collection of statements)

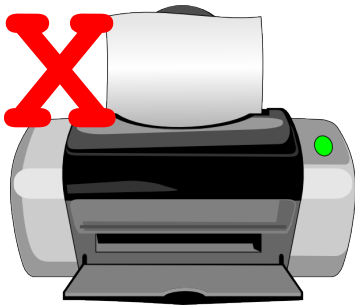
The method called "main" is where **execution begins** in any Java program!

Java programs are made up of **class definitions**

```
public class Hello{
  public static void main(String[] args) {
     System.out.println("CS170 Rocks!");
  }
}
```

"print" in the context of programming…

…usually means displaying something on the screen (a.k.a *console* or *terminal*)



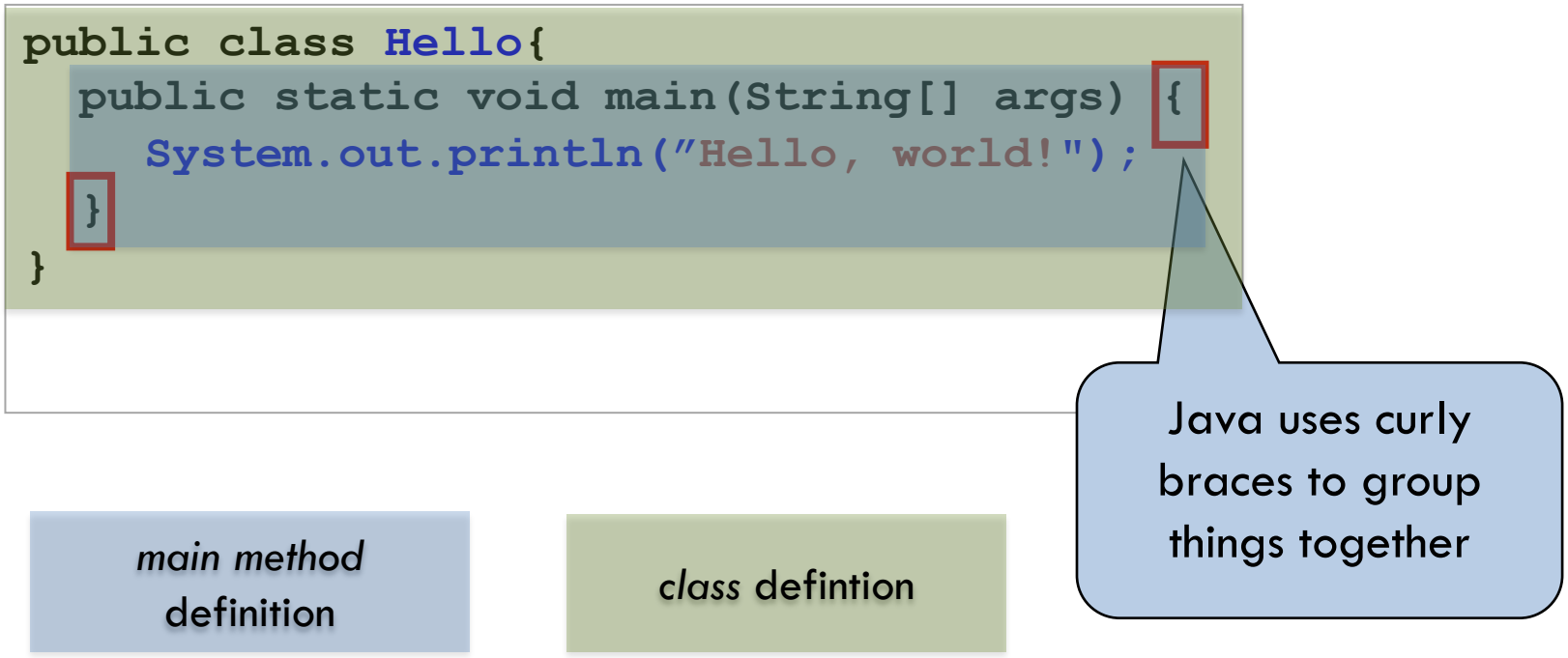
What's "System.out.println(…)"

```
public class Hello{
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

A method provided by a **Java library**

Java Library: Collection of class and method definitions

Blocks of code {…}



What if we wanted to add a **comment**?

```
public class Hello{
  public static void main(String[] args) {
     // generate some simple output
     System.out.println("Hello, world!");
  }
}
```

// program comments
Text you put in a program
(usually to explain what it does)
but is ignored by the compiler

Some sample comments

// I am not sure if we need this, but too scared to delete!

// This method needs to be completed.

// The sub() method returns subtraction of given numbers

// above comment is lexically equivalent to the following
// multiplies pi by 4

https://loudprogrammer.net/best-comments-in-source-code-i-ever-encountered/

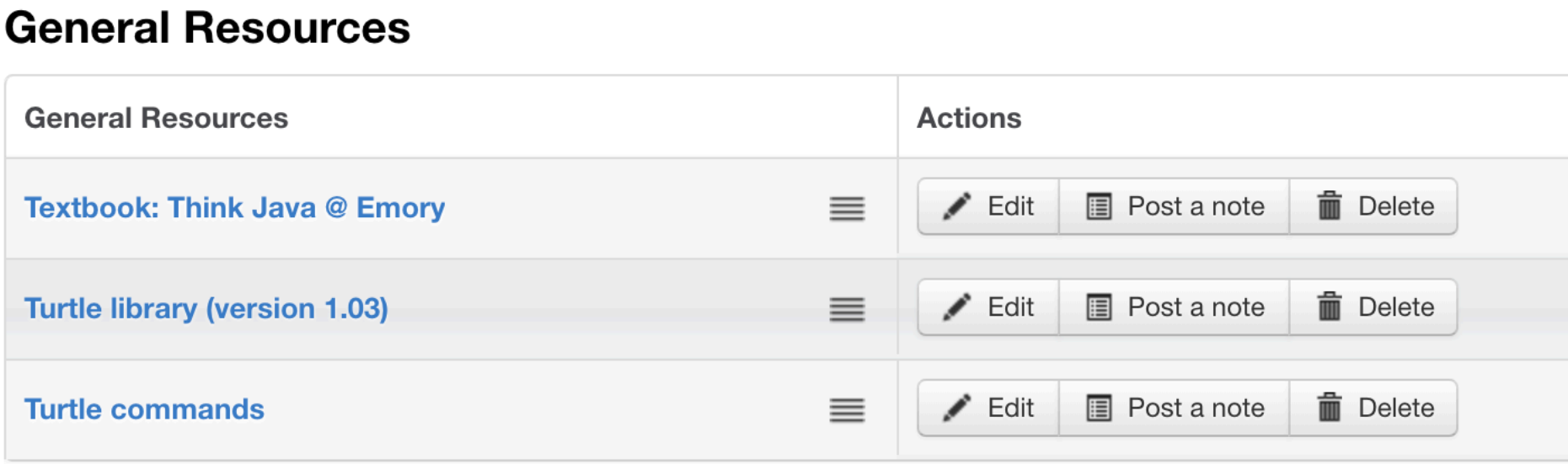
Let's write our first Java program...

https://loudprogrammer.net/best-comments-in-source-code-i-ever-encountered/

Turtle Graphics Example

<title>code ninja</title>

Turtle Library is available on Piazza => Resources



Turtle Graphics

To use Turtle Graphics commands, copy the `Turtle.java` library file in the same folder as your program.

Command	Description	Example
new Turtle()	Creates a new turtle.	`Turtle t = new Turtle();`
forward(steps)	Moves the turtle forward.	`t.forward(100);`
backward(steps)	Moves the turtle backward.	`t.backward(100);`
left(degrees)	Rotates the turtle counter-clockwise.	`t.left(90);`
right(degrees)	Rotates the turtle clockwise.	`t.right(90);`
penup()	Raises the turtle's pen. When the pen is up, the turtle does not draw lines on the screen.	`t.penup();`
pendown()	Lowers the turtle's pen. When the pen is down, the turtle draws lines on the screen.	`t.pendown();`
color(colorName)	Sets the color of the turtle.	`t.color("red");`
color(red, green, blue)	Sets the color of the turtle, specifying the individual red, green, and blue components of the color. 0 <= red <= 255 0 <= green <= 255 0 <= blue <= 255	`t.color(240, 170, 65);`
delay(milliseconds)	Sets the turtle's delay between moves. The lower the delay, the faster the turtle. Default: 100 ms.	`t.delay(20);`
hideturtle()	Hides the turtle so it is not visible on the screen.	`t.hideturtle();`
showturtle()	Shows the turtle so it is visible on the screen.	`t.showturtle();`