

# CSE 121 Lesson 3: Variables, Strings, Debugging

Matt Wang  
Spring 2024



TAs:	Andy	Anju	Archit	Arkita	Autumn	Christian
	Hannah H	Hannah S	Heather	Hibbah	Janvi	Jessie
	Jonus	Julia	Luke	Maria	Mia	Ritesh
	Shayna	Simon	Trey	Vidhi	Vivian	Gumball?

[sli.do #cse121-3](https://sli.do/#cse121-3)

Today's playlist:  
[CSE 121 lecture beats 24sp](#)

# Announcements, Reminders

- P0 was released on Thu and is due Wed, Apr 10<sup>th</sup>
- Quiz 0 scheduled for Apr 25<sup>th</sup> (about 3 weeks away)
  - More details will be released in the coming week!
  - Prep includes practice quizzes, sections, etc.
- Quick demo: [Ed shortcuts page on website](#)

# (PCM) Variables – Declaration, Initialization

- Now that we know about different types and data, we can learn about how to store it!
- Java allows you to create variables within a program. A variable has
  - A type
  - A name
  - (Potentially) a value it is storing

Declaration:     `int x;`  
Initialization:   `x = 30;`

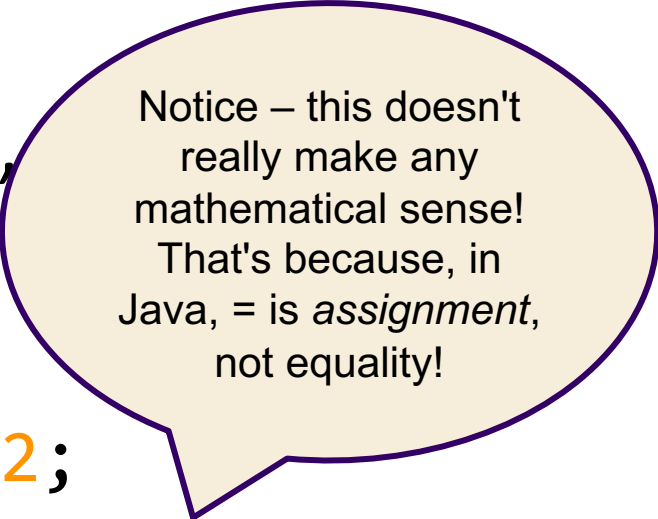
Or all in one line:

```
int x = 30;
```

# (PCM) Variables – Manipulation

They're made to be manipulated, modified,

```
int myFavoriteNumber = 7;  
int doubleFV = myFavoriteNumber * 2;  
myFavoriteNumber = myFavoriteNumber + 3;
```



Notice – this doesn't really make any mathematical sense! That's because, in Java, = is *assignment*, not equality!

# New Operators! (1/3)

```
myFavoriteNumber = myFavoriteNumber + 3;
```

This pattern is so common, we have a shorthand for it!

```
myFavoriteNumber += 3;
```

Note: this works for both numeric addition and string concatenation!

# New Operators! (2/3)

The shorthands `-=`, `*=`, `/=`, and `%=` exist too!

Take an educated guess: what do you think they do?

```
myFavoriteNumber /= 3;
```

Should this work for integers? Doubles? Strings?

# New Operators! (3/3)

There are even shorter operators for “incrementing” and “decrementing”!

```
myFavoriteNumber++; // equal to myFavoriteNumber += 1;  
myFavoriteNumber--; // equal to myFavoriteNumber -= 1;
```

Should this work for integers? Doubles? Strings?

# Poll in with your answer!



[sli.do #cse121-3](https://sli.do/#cse121-3)

What do a, b, and c hold after this code is executed?

```
int a = 10;  
int b = 30;  
int c = a + b;  
c -= 10;  
a = b + 5;  
b /= 2;
```

A. 10, 30, 40


B. 35, 15, 30

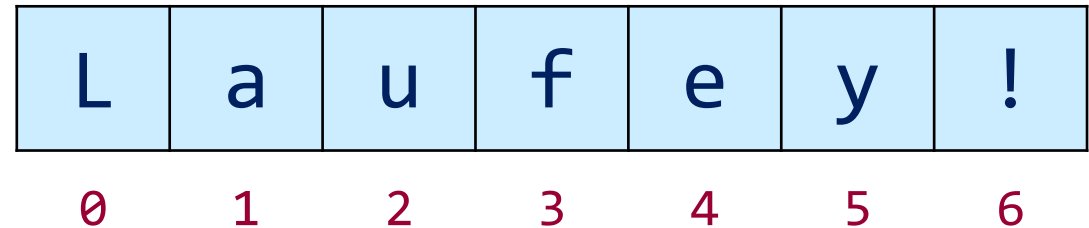
C. 35, 15.5, 30

D. 20, 15, 30



# (PCM) Strings and chars

- String = sequence of characters treated as one, yet can be indexed to get individual parts
- Zero-based indexing 



- **Side note:** new data type!  
char, represents a single character,  
so we use single quotes  
Strings are made up of chars!

# (PCM) String Methods

Usage: `<string variable>.<method>(…)`

Method	Description
<code>length()</code>	Returns the length of the string.
<code>charAt(i)</code>	Returns the character at index <i>i</i> of the string
<code>indexOf(s)</code>	Returns the index of the first occurrence of <i>s</i> in the string; returns -1 if <i>s</i> doesn't appear in the string
<code>substring(i, j)</code> or <code>substring(i)</code>	Returns the characters in this string from <i>i</i> (inclusive) to <i>j</i> (exclusive); if <i>j</i> is omitted, goes until the end of the string
<code>contains(s)</code>	Returns whether or not the string contains <i>s</i>
<code>equals(s)</code>	Returns whether or not the string is equal to <i>s</i> (case-sensitive)
<code>equalsIgnoreCase(s)</code>	Returns whether or not the string is equal to <i>s</i> ignoring case
<code>toUpperCase()</code>	Returns an uppercase version of the string
<code>toLowerCase()</code>	Returns a lowercase version of the string

# Poll in with your answer!



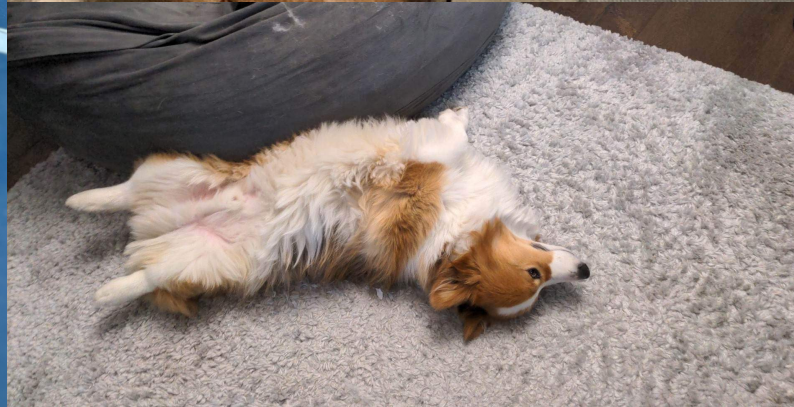
[sli.do #cse121-3](https://sli.do/#cse121-3)

Suppose `s` contains the String "bubble gum". Which option below would result in `s` containing "Gumball" instead?

b	u	b	b	l	e		g	u	m
0	1	2	3	4	5	6	7	8	9

- A. `s.substring(7) + "ball";`
- B. `s = s.substring(7, 9) + "ball";`
- C. `s = s.charAt(7).toUpperCase() + "ball";`
- D. `s = s.substring(7, 8).toUpperCase() + s.substring(8) + "ball";`

# Interlude: Gumball





# Food for Thought



A weekly section where I introduce open problems related to our lecture topic(s) of the week.

Goals:

1. give you “conversational familiarity” with CS terminology
2. see how CS interacts with other fields and people!
3. point you in the direction of more CSE (or adjacent) classes

Note: not tested content. Just food for thought :)

# What's in a (variable) name or String?

Switch over to [Ed](#) and do some experiments (with a partner)! Then, report back on [sli.do](#).

1. What counts as one character?
2. What kinds of characters are “allowed” in Strings?
3. What kinds of characters are “allowed” in variable names?
4. Are the lengths of the Strings what you expect?  
Why or why not?



[sli.do #cse121-3](#)

# Dessert for Thought!

This is the beginning of a very interesting rabbit hole! But also, a decision made by the Java designers.

You will also make decisions like these!

- for example, what is a “valid name”?
- something to reflect on as you learn more about CS...