

LEC 06

CSE 121

Cumulative Sum, Scope, Class Constants

Questions during Class?

Raise hand or send here

sli.do #cse121



BEFORE WE START

Talk to your neighbors:

What's your favorite hobby?

Music: [CSE 121 26wi Lecture Tunes](#)

Instructor: Miya Natsuvara

TAs:	Amogh	Hayden	Anum	Sam	Shayna
	William	Aki	Abdul	Ethan	Jesse
	Johnathan	Spencer	Janvi	Jessica	Minh
	Anant	Savannah	Navya	Paul	Cayden
	Reese	Tamsyn	Ruslana	Carson	

Agenda

- **Announcements, Reminders** 
- Random warmup
- Math Facts recap
- Scope, class constants review
- Code examples
 - Cumulative sums!

Announcements, Reminders

- Resubmission Cycle 0 (R0) due tomorrow, Thursday, Jan 29th
- P1: Election Simulator out today, due Tuesday, Feb 3rd
- Quiz 0 is **tomorrow**, Thursday, Jan 29th (in your registered quiz section)
 - can't make it? email Miya ASAP!!

A bit more on Quiz 0

- Taken on paper, in your registered quiz section
 - Quiz logistics in detail
- Broadly: focused on concepts, reading, writing, and debugging code
- Main topics: printing, datatypes, expressions, variables, Strings, for loops
- You get to bring one sheet of notes (8.5x11in, or standard A4)
- See Ed for provided reference sheet, practice quizzes, and practice quiz solutions

A bit more on P1

1. this is a big jump from C1. **Start early!**
2. read the spec carefully, take notes
 - make sure you understand what you're doing *before* you start coding
3. there will be another development slide to help you do *iterative development*, along with a recommended development strategy!
 - you don't have to follow it, but we recommend making *some* plan for how to approach the project (since it's more substantial than previous projects!)

Agenda

- Announcements, Reminders
- **Random warmup** ←
- Math Facts recap
- Scope, class constants review
- Code examples
 - Cumulative sums!

PCM: Random

The Random class helps us generate pseudo-random numbers.

- We create a new Random number generator by calling `new Random()`
- First, we need to **import** the Random class with `import java.util.*;`
- We can “seed” the generator to make it behave *deterministically*

Method	Description
<code>nextInt()</code>	Returns a random integer
<code>nextInt(max)</code>	Returns a random integer in the range $[0, max]$, or in other words, 0 to $max-1$ inclusive
<code>nextDouble()</code>	Returns a random double in the range $[0.0, 1.0]$



Practice: Think



sli.do #cse121

Assuming you've declared: `Random randy = new Random();`

Which of these best models picking a random card? (number between 1 and 13 inclusive)

- A. `randy.nextInt()`
- B. `randy.nextInt(13)`
- C. `randy.nextInt(13) + 1`
- D. `randy.nextInt(13 + 1)`



Practice: Think



sli.do #cse121

Assuming you've declared: `Random randy = new Random();`

Which of these best models picking a random card? (number between 1 and 13 inclusive)

- A. `randy.nextInt()`
- B. `randy.nextInt(13)`
- C. `randy.nextInt(13) + 1`
- D. `randy.nextInt(13 + 1)`

Agenda

- Announcements, Reminders
- Random warmup
- **Math Facts recap** 
- Scope, class constants review
- Code examples
 - Cumulative sums!

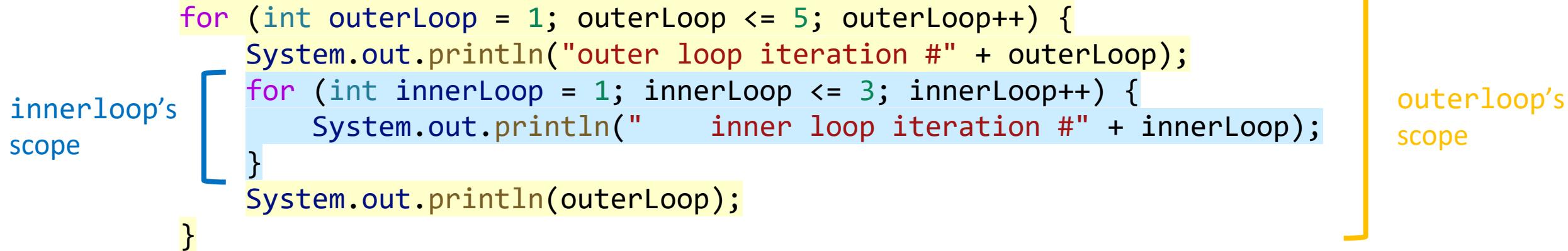
Agenda

- Announcements, Reminders
- Random warmup
- Math Facts recap
- **Scope, class constants review** 
- Code examples
 - Cumulative sums!

PCM: Scope

Scope: the part of a program where a variable exists (and can thus be referenced, modified, or used).

- General Rule: a variable ‘exists’ from its **declaration** to the **closing brace of that indentation level }**



```
for (int outerLoop = 1; outerLoop <= 5; outerLoop++) {  
    System.out.println("outer loop iteration #" + outerLoop);  
    for (int innerLoop = 1; innerLoop <= 3; innerLoop++) {  
        System.out.println("    inner loop iteration #" + innerLoop);  
    }  
    System.out.println(outerLoop);  
}
```

innerloop's scope [] outerloop's scope]

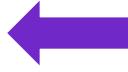
PCM: Class Constants

A fixed value visible (in-scope) to the whole program (the entire *class*).

Value is set at declaration, **cannot** be reassigned – value is *constant*.

```
public static final type NAME_OF_CONSTANT = expression;
```

Agenda

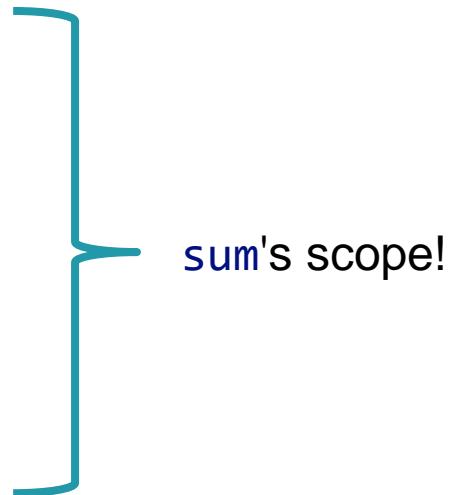
- Announcements, Reminders
- Random warmup
- Math Facts recap
- Scope, class constants review
- **Code examples** 
 - Cumulative sums!

New: Cumulative Sum pattern

Where we *accumulate* information into a variable while a loop is running!

- In order to this, we need to declare the variable *outside* of the loop

```
int sum = 0;  
for (int i = 1; i <= 10; i++) {  
    sum += i;  
}  
System.out.println(sum);
```



sum's scope!

What is the scope of **sum**?

Addition Problem Development Strategy / Pseudocode