LEC 06

CSE 121

Cumulative Sum, Scope, Class Constants

BEFORE WE START

Talk to your neighbors:

What's your favorite hobby?

Music: CSE 121 26sp Lecture Tunes

Instructor: Matt Wang

TAs:

Abdul	Amogh	Anant	Anum	Cayden
Dalton	Ethan	Hayden	Jesse	Jessica
Johnathan	Minh	Navya	Paul	Reese
Ruslana	Sam	Savannah	Spencer	Shayna
Tamsyn	TJ	Trey		

Questions during Class?

Raise hand or send here

sli.do #cse121

Agenda (1/5)

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

Announcements, Reminders

- Resubmission Cycle 0 (R0) due tomorrow, Thursday, April 23rd
- P1: Election Simulator out today, due Tuesday, April 28th
- Quiz 0 is ***tomorrow***, Thursday, April 23rd (in your registered quiz section)
  - can't make it? email Matt ASAP!!

A bit more on Quiz 0

- Taken on paper, in your registered quiz section
  - see Ed post on 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)
- Ed resources tab has reference sheet, practice quizzes, and practice quiz key
  - Matt fixed a mistake in Practice Quiz 0 Version 1 Q2 last night!

A bit more on P1

1. this is a big jump from C1. **Start early!**
2. read the spec carefully (maybe take notes!)
   - make sure you understand what you're doing *before* you code
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 (2/5)

- Announcements, Reminders
- **Random warmup** ⟵
- Math Facts continued
- 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
`nextInt()`	Returns a random integer
`nextInt(max)`	**Returns a random integer in the range [0, *max*), or in other words, 0 to *max*-1 inclusive**
`nextDouble()`	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 (3/5)

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

Agenda (4/5)

- Announcements, Reminders
- Random warmup
- Math Facts continued
- **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 }**
- Exception applies to for loops (header is "part" of the body)

```
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 (5/5)

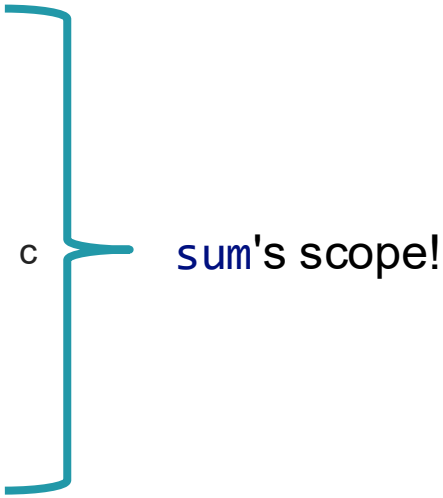
- Announcements, Reminders
- Random warmup
- Math Facts continued
- Scope, class constants review
- **Code examples** ⟵
  - **Cumulative sums!**

New: Cumulative Sum pattern

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

- In order to do 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);
```



What is the scope of `sum`?