LEC 06

CSE 121

Methods & Parameters

Questions during Class?

Raise hand or send here

sli.do #cse121

BEFORE WE START

Talk to your neighbors:

What's your favourite study spot on campus? Off campus?

Music: 121 25wi lecture playlist

Instructor: Matt Wang

TAs: Ailsa, Alice, Chloë, Christopher, Ethan, Hanna, Hannah, Hibbah, Janvi, Judy, Julia, Kelsey, Lucas, Luke, Maitreyi, Merav, Ruslana, Samrutha, Sam, Shayna, Sushma, Vivian

Announcements, Reminders

- Resubmission Cycle 0 (R0) due tomorrow, Thursday January 30
- P1: Election Simulator out today, due Tuesday February 4
- Quiz 0 is on Thursday, February 6 (in your registered quiz section)
  - can't make it? email Matt ASAP
- Reminder: Ed Shortcuts page!
- Support reminders:
  - Matt's OHs: Mon 2:30 – 3:20, Wed 3:30 – 4:20, Fri 1:30 – 2:20
  - IPL: Mon-Thu 12:30 – 9:30, Fri 12:30 – 5:30, Sat 1:30 – 3:30
  - Async via Ed & email!

A bit more on Quiz 0

- taken on your computer, in your quiz section
- broadly: focused on concepts, reading, and debugging code
- covers material up to today's lecture (methods & parameters) but no further (e.g. no returns)

Expect tomorrow & Friday, two Ed announcements:

1. the full quiz policy (the "cover sheet" of an exam)
2. two practice quizzes!

Will discuss more on Friday (including study tips!)

A bit more on P1

1. this is a big jump from C1. **Start early!**
2. P1 does *not* require you to use methods (though you can!).
   - advice: feeling shaky on writing methods? don't do it for P1
3. advice: don't put off P1 to study for Quiz 0
   - easy way to fall behind in the class
   - P1's technical topics are nested for loops, scope, and Random. **These are all on your Quiz!** Doing P1 *is* studying!

Last Time: Nested For Loops

Reviewed syntax & conventions (e.g. `i-j-k` naming)

- advice: don't think of nested for loops as anything "special"
- advice: break nested loops up into smaller problems, do one at a time

```
for (int outerLoop = 1; outerLoop <= 5; outerLoop++) {
    System.out.println("outer loop iteration #" + outerLoop);
    for (int innerLoop = 1; innerLoop <= 7; innerLoop++) {
        System.out.println("    inner loop iteration #" + innerLoop);
    }
    System.out.println(outerLoop);
}
```

Last Time: Random

A `Random` **object** generates *pseudo-random* numbers.

`nextInt(max)` returns a pseudo-random `int` value from `[0, max)`
i.e. between `0` and `max-1`

```
Random rand = new Random();
```

type — name — Random creation code

```
rand.nextInt(6) + 1
```

Practice: Think

sli.do #cse121

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

Which of these best models picking a random card? (1-13 inclusive)

A. `randy.nextInt()`

B. `randy.nextInt(13)`

C. `randy.nextInt(13) + 1`

D. `randy.nextInt(14)`

Practice: Pair

sli.do #cse121

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

Which of these best models picking a random card? (1-13 inclusive)

A. `randy.nextInt()`

B. `randy.nextInt(13)`

C. `randy.nextInt(13) + 1`

D. `randy.nextInt(14)`

Last Time: Math

Calling:

`Math.<method>(…)`

Method	Description
`Math.abs(`*value*`)`	Returns the absolute value of *value*
`Math.ceil(`*value*`)`	Returns *value* rounded up
`Math.floor(`*value*`)`	Returns *value* rounded down
`Math.max(`*value1*`, `*value2*`)`	Returns the larger of the two values
`Math.min(`*value1*`, `*value2*`)`	Returns the smaller of the two values
`Math.round(`*value*`)`	Returns *value* rounded to the nearest whole number* note: need to cast result to int (it's complicated!)
`Math.sqrt(`*value*`)`	Returns the square root of *value*
`Math.pow(`*base*`, `*exp*`)`	Returns *base* raised to the *exp* power

PCM: Methods

Writing our own **methods** allows us to define our own commands!

(naming conventions for methods are same as variables: `camelCased`)

```
public static void myMethod() {
  /***
  Your code here
  **/
}
```

Practice: Think

sli.do #cse121

```
public class HelloGoodbye {
    public static void main(String[] args) {
        welcome();
        hello();
        goodbye();
    }

    public static void hello() {
        System.out.print("Hello! ");
        glad();
    }

    public static void goodbye() {
        System.out.println("Goodbye!");
    }

    public static void welcome() {
        System.out.print("Welcome! ");
        glad();
    }

    public static void glad() {
        System.out.println("Glad you're here.");
    }
}
```

What is the output of this program?

A.
```
Welcome! Glad you're here.
Hello! Glad you're here.
Goodbye!
```

B.
```
Welcome!
Hello!
Goodbye!
```

C.
```
Welcome! Hello! Goodbye!
```

D.
```
Welcome!
Glad you're here.
Hello!
Glad you're here.
Goodbye!
```

Practice: Pair

sli.do #cse121

```
public class HelloGoodbye {
    public static void main(String[] args) {
        welcome();
        hello();
        goodbye();
    }

    public static void hello() {
        System.out.print("Hello! ");
        glad();
    }

    public static void goodbye() {
        System.out.println("Goodbye!");
    }

    public static void welcome() {
        System.out.print("Welcome! ");
        glad();
    }

    public static void glad() {
        System.out.println("Glad you're here.");
    }
}
```

What is the output of this program?

A.
```
Welcome! Glad you're here.
Hello! Glad you're here.
Goodbye!
```

B.
```
Welcome!
Hello!
Goodbye!
```

C.
```
Welcome! Hello! Goodbye!
```

D.
```
Welcome!
Glad you're here.
Hello!
Glad you're here.
Goodbye!
```

PCM: Parameters

Definition: a value passed to a method by its caller. “Like” a variable!

```
public static void myMethod(String musicalAct) {
    System.out.print(musicalAct + " is the best!");
    ...
}
```

Calling a method with a parameter…

```
myMethod("Laufey"); // prints: Laufey is the best!
```

PCM: Scope, Redux

Our scope rules also apply to methods and parameters!

- General rule: from its **declaration to the next closing brace, }**
- a variable declared in a method only exists <u>in that method</u>!

```
public static void example(int n) {
    System.out.println("hello");
    int x = 3;
    for (int i = 1; i <= n; i++) {
        System.out.print(x);
    }
}
```

i's scope

x's scope

n's scope

New: 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;
```

New: Method Comments

Each method you write (except main) should have a short comment!

```
// Randomly generates an addition problem where the
// operands are in the range 1-10 (inclusive),
// and prints the result, rounded to two decimal places.
public static void addTwoRandomNumbers() {
    Random randy = new Random();
    int num1 = randy.nextInt(10) + 1;
    int num2 = randy.nextInt(10) + 1;
    int sum = num1 + num2;
    ...
}
```

Announcements, Reminders (again)

- Resubmission Cycle 0 (R0) due tomorrow, Thursday January 30
- P1: Election Simulator out today, due Tuesday February 4
- Quiz 0 is on Thursday, February 6th (in your registered quiz section)
  - can't make it? email Matt ASAP
- Reminder: Ed Shortcuts page!
- Support reminders:
  - Matt's OHs: Mon 2:30 – 3:20, Wed 3:30 – 4:20, Fri 1:30 – 2:20
  - IPL: Mon-Thu 12:30 – 9:30, Fri 12:30 – 5:30, Sat 1:30 – 3:30
  - Async via Ed & email!