CSE 121 Lesson 6:
Methods & Parameters

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 #cse121-6

Today's playlist:
CSE 121 lecture beats 24sp

Announcements & Reminders

- Resubmission Cycle 0 (R0) due tomorrow, Apr 17
- Programming Assignment 1 (P1) out today, due Apr 23
  - note: a big jump from C1. start early!
- Quiz 0 next week in section (Thursday, Apr 25)
  - Topics: everything up to and including today's lecture
  - Optional review session in JHN 102 on Tuesday, April 23
  - Try practice quizzes, starred (⭐) section problems
  - More review in upcoming sections & lectures

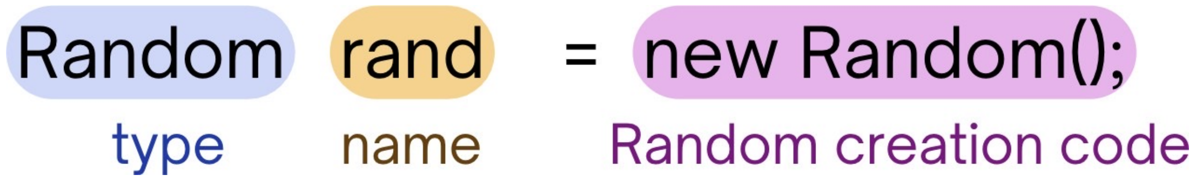
Last Time 1

- Nested for loops
  - Syntax & conventions: (i , j , k)
  - Applications: “doing the same thing for multiple iterations”

```
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 2

- Random
  - A Random object generates *pseudo*-random numbers
  - `nextInt(max)` returns random `int` value [0, max)
    i.e. between 0 and max-1



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

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

(PCM) Methods

Writing our own **methods** allow us to define our own statements / commands in Java!

- Naming conventions for methods are the same as variables: `camelCased`

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

Poll in with your answer!

sli.do #cse121-6

```java
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!
```

Class Constants

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

- Value can be set only at declaration; **cannot** be reassigned (so the value is constant)

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

Method Comments!

- Now that we know how to write methods, we have a new form of documentation (using comments) to write.
- Each method you write (except for main) should be accompanied by a short comment that describes what it does.

```java
// 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;
    ...
}
```

(PCM) Parameters

Definition: A value passed to a method by its caller

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

Calling a method with a parameter…

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

Scope 1

- Definition: The part of a program where a variable exists (and can thus be referenced/modified/used).
  - From its **declaration to the end of the { } braces**
  - Ex: a variable declared in a `for` loop only exists in that loop!

```
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

Scope 2

- Definition: The part of a program where a variable exists (and can thus be referenced/modified/used).
  - From its **declaration to the end of the { } braces**
  - Ex: a variable declared in a method exists only in that method!

```java
public static void example() {
    System.out.println("hello");
    int x = 3;
    for (int i = 1; i <= 10; i++) {
        System.out.print(x);
    }
}
```

i's scope

x's scope