CSE 121 – Lesson 8

Elba Garza & Matt Wang

Winter 2024

sli.do #cse121-8

TAs:	Abby	Aishah	Anju	Annie	Archit	Ayesha	Christian
	Hannah	Heather	Hibbah	Jacob	James	Janvi	Jasmine
	Jonus	Julia	Lucas	Luke	Maria	Nicole	Shananda
	Shayna	Trey	Vidhi	Vivian			

Today's playlist:
CSE 121 24wi lecture beats :D

Announcements, Reminders

- Alt Text video walk-through now available
- Programming Assignment 1 (P1) was due last night!
- Creative Project 2 (C2) releasing later today (due Feb 6th)
- Resubmission Cycle 1 (R1) due tomorrow, Feb 1st by 11:59pm
- Resubmission Cycle 2 (R2) opens tomorrow (due Feb 8th)
  - Eligible assignments: C0, P0, C1, P1
- Wednesday Feb 7th: Mid-term Formative Feedback in class
- Quiz 0 is tomorrow in your quiz section!
  - Quiz logistics announced on Ed

Common Problem-Solving Strategies

- **Analogy** – Is this similar to another problem you've seen?
- **Brainstorming** – Consider steps to solve problem before jumping into code
  - Try to do an example "by hand" → outline steps
- **Solve sub-problems** – Is there a smaller part of the problem to solve?
- **Debugging** – Does your solution behave correctly?
  - What is it doing?
  - What do you expect it to do?
  - What area of your code controls that part of the output?
- **Iterative Development** – Can we start by solving a different problem that is easier?

(Recall) Methods & 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("Olivia Rodrigo"); // Prints out
                            // "Olivia Rodrigo is
                            // the best!"
```

(Recall) Returns 1

Returns allow us to send values out of a method

```
public static <type> myMethod(<zero or more params>) {
    ...
    return <value of correct type>
}
```

Evaluates the expression

Returns this value to where the method is called from

Method immediately exits!

Calling a method that returns a value…

```
<type> result = myMethod(...);
```

(Recall) Returns 2

Returns allow us to send values out of a method

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

Calling a method with a parameter…

```
String s = myMethod("Olivia Rodrigo"); // Prints and returns
                                       // "Olivia Rodrigo is
                                       // the best!"
```

(Recall) Returns 3

Returns allow us to send values out of a method

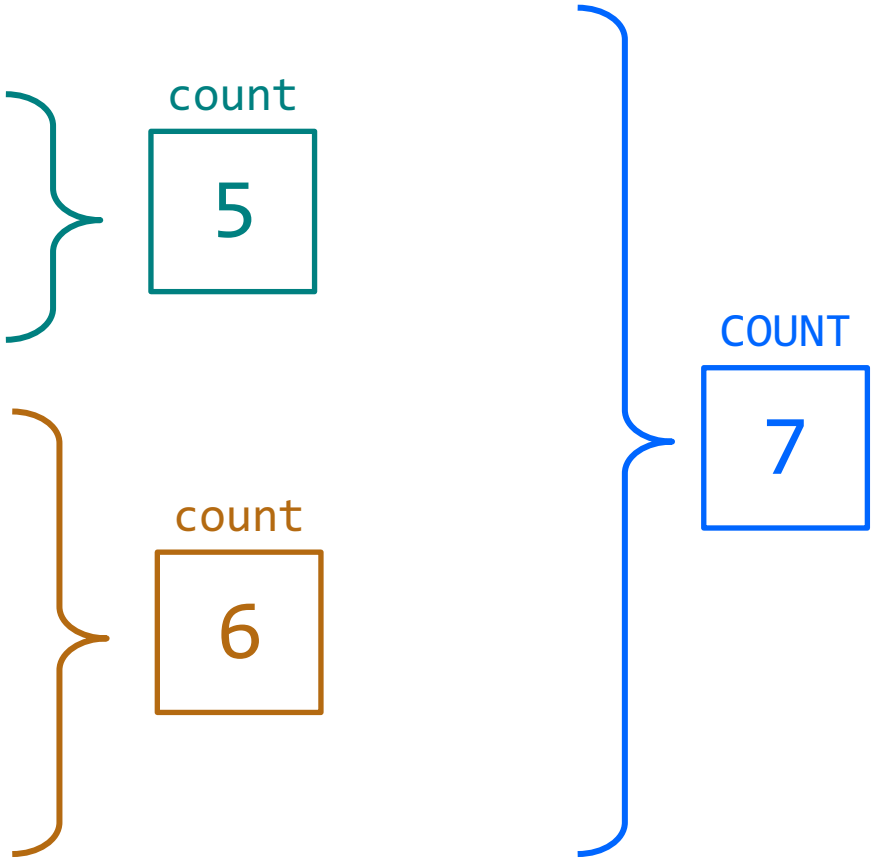
```
public static String myMethod(String musicalAct) {
    ...
    return musicalAct + " is the best!”
}
```

Calling a method with a parameter…

```
String s = myMethod("Olivia Rodrigo"); // Returns
                                       // "Olivia Rodrigo is
                                       // the best!"
```

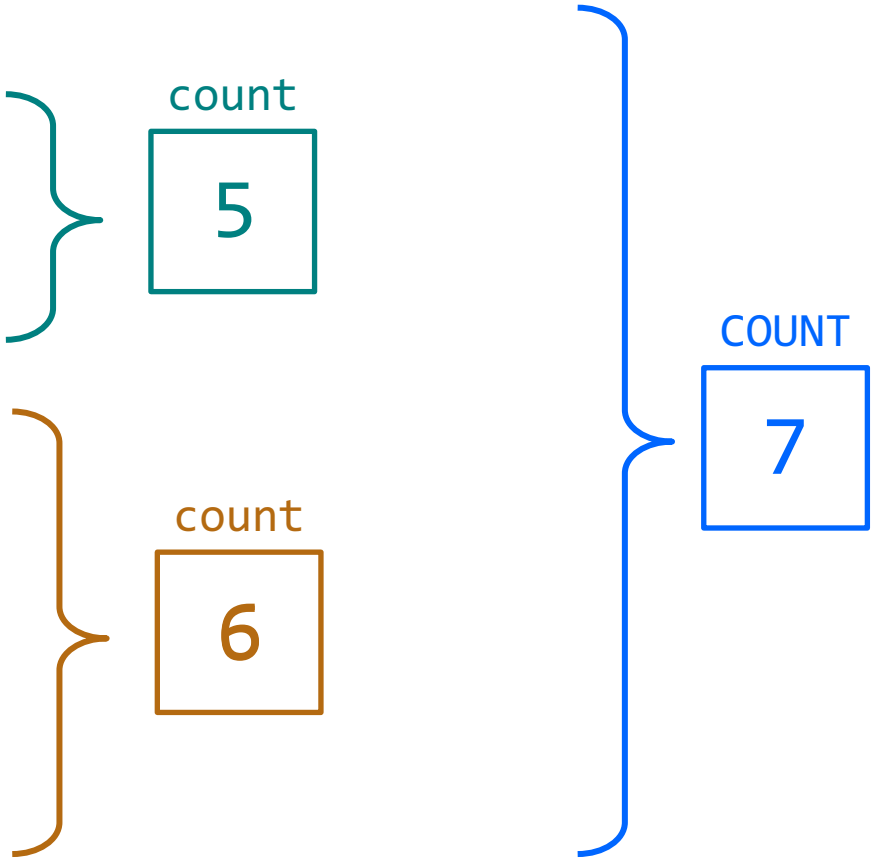
(Recall) Tricky Poll 1: Last line printed?

```
public static final int COUNT = 7;
public static void main(String[] args) {
  int count = 5;
  line(count);
  System.out.println("count is: " + count);
}

public static void line(int count) {
  for (int i = 1; i <= count; i++) {
    System.out.print("*");
  }
  count++;
  System.out.println();
}
```



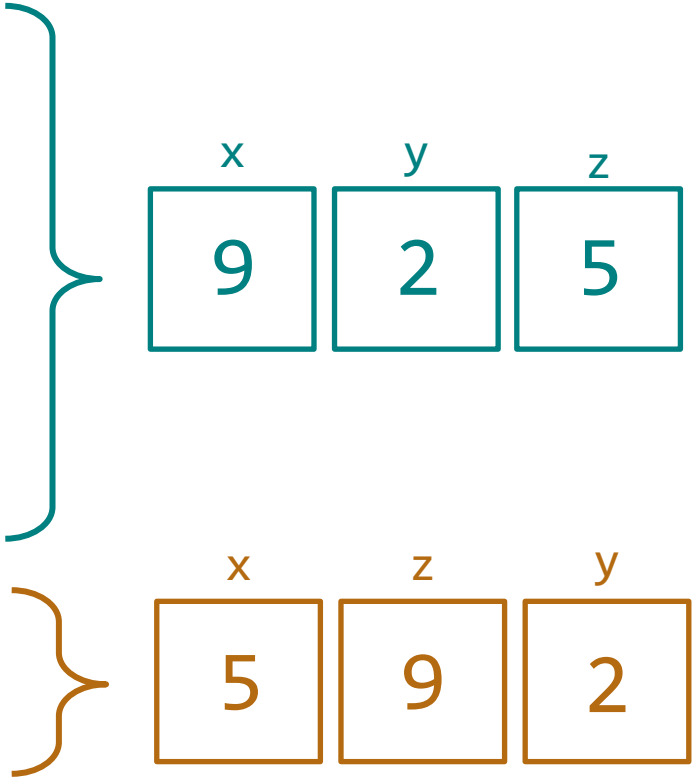
Tricky Poll 1: Returnable

```
public static final int COUNT = 7;
public static void main(String[] args) {
  int count = 5;
  count = line(count);
  System.out.println("count is: " + count);
}

public static int line(int count) {
  for (int i = 1; i <= count; i++) {
    System.out.print("*");
  }
  count++;
  System.out.println();
  return count;
}
```



Tricky Poll 2: x's, and y's, and z's, oh my.

```
public static void main(String[] args) {
  int x = 9;
  int y = 2;
  int z = 5;

  mystery(z, y, x);

  mystery(y, x, z);
}

public static void mystery(int x, int z, int y) {
  System.out.println(z + " and " + (y - x));
}
```



Example of returns: Math class

Methods	Returns
`Math.abs(`*value*`)`	Absolute value of *value*
`Math.ceil(`*value*`)`	*value* rounded up
`Math.floor(`*value*`)`	*value* rounded down
`Math.max(`*value1*`, `*value2*`)`	Larger of the two given values
`Math.min(`*value1*`, `*value2*`)`	Smaller of the two given values
`Math.round(`*value*`)`	*value* rounded to the nearest whole number
`Math.sqrt(`*value*`)`	Square root of *value*
`Math.pow(`*base*`, `*exp*`)`	*base* to the *exp* power

Math example

```
double value = 823.577564893;
double roundedValue = (double) Math.round(value * 100) / 100;
```

Poll in with your answer!

sli.do #cse121-8

What is the correct implementation of a `maxDatingAge` method?

A.
```
public static void maxDatingAge(int age) {
    int maxDatingAge = age - 7 * 2;
    return maxDatingAge;
}
```

B.
```
public static void maxDatingAge(int age) {
    int maxDatingAge = age - 7 * 2;
}
```

C.
```
public static int maxDatingAge(int age) {
    int maxDatingAge = (age - 7) * 2;
    return maxDatingAge;
}
```

D.
```
public static int maxDatingAge(int age) {
    return (age - 7) * 2;
}
```