CSE 121 Lesson 7: Methods, Parameters, Returns
Elba Garza & Matt Wang
Winter 2024

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

sli.do #CSE121-7

Today’s playlist: CSE 121 24wi lecture beats 😁
Announcements, Reminders

• Programming Assignment 1 is out, due Tues Jan 30
  • Start early! This one is tough!

• Resubmission Cycle 1 released yesterday, due Thurs Feb 1

• Quiz 0 is Thurs Feb 1!
  • Next week’s section content is focused on prepping you for the quiz!

• New [Ed Shortcuts page](#) on class website!
**(Review) Parameters**

Definition: A value passed to a method by its caller

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

Calling a method with a parameter...
```
myMethod("Olivia Rodrigo"); // Olivia Rodrigo is the best!
```
(Review) Scope

The part of a program where a variable exists.

• From its declaration to the end of the \{ \} braces
• Ex: a variable declared in a for loop only exists in that loop
• 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
(Review) 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)

```java
public static final type NAME_OF_CONSTANT = expression;
```
Poll in with your answer!

What will be the last line of output after this code has executed?
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();
}

A. count is: 1
B. count is: 5
C. count is: 6
D. count is: 7
What is the output of this program?

```java
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));
}
```

A. 2 and 4
    9 and 3

B. 5 and -7
    5 and -7

C. 9 and -3
    5 and -7

D. I'm lost
(PCM) Returns

Returns allow us to send values out of a method

```java
public static <type> myMethod(int num) {
    System.out.print(num + " is the best!");
    ...
    return <value of correct type>
}
```

Calling a method that returns a value...

```java
<type> result = myMethod(42);
```

Evaluates the expression

Returns this value to where the method is called from

Method immediately exits
## (Recall) String Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>length()</td>
<td><strong>Returns</strong> the length of the string.</td>
</tr>
<tr>
<td>charAt(i)</td>
<td><strong>Returns</strong> the character at index $i$ of the string</td>
</tr>
<tr>
<td>indexOf(s)</td>
<td><strong>Returns</strong> the index of the first occurrence of $s$ in the string; returns -1 if $s$ doesn't appear in the string</td>
</tr>
<tr>
<td>substring($i, j$) or substring($i$)</td>
<td><strong>Returns</strong> the characters in this string from $i$ (inclusive) to $j$ (exclusive); if $j$ is omitted, goes until the end of the string</td>
</tr>
<tr>
<td>contains(s)</td>
<td><strong>Returns</strong> whether or not the string contains $s$</td>
</tr>
<tr>
<td>equals(s)</td>
<td><strong>Returns</strong> whether or not the string is equal to $s$ (case-sensitive)</td>
</tr>
<tr>
<td>equalsIgnoreCase(s)</td>
<td><strong>Returns</strong> whether or not the string is equal to $s$ ignoring case</td>
</tr>
<tr>
<td>toUpperCase()</td>
<td><strong>Returns</strong> an uppercase version of the string</td>
</tr>
<tr>
<td>toLowerCase()</td>
<td><strong>Returns</strong> a lowercase version of the string</td>
</tr>
</tbody>
</table>

Usage: `<string variable> . <method>(...)`
## Example of returns: Math class

<table>
<thead>
<tr>
<th>Methods</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math.abs(value)</td>
<td>Absolute value of value</td>
</tr>
<tr>
<td>Math.ceil(value)</td>
<td>value rounded up</td>
</tr>
<tr>
<td>Math.floor(value)</td>
<td>value rounded down</td>
</tr>
<tr>
<td>Math.max(value1, value2)</td>
<td>Larger of the two given values</td>
</tr>
<tr>
<td>Math.min(value1, value2)</td>
<td>Smaller of the two given values</td>
</tr>
<tr>
<td>Math.round(value)</td>
<td>value rounded to the nearest whole number</td>
</tr>
<tr>
<td>Math.sqrt(value)</td>
<td>Square root of value</td>
</tr>
<tr>
<td>Math.pow(base, exp)</td>
<td>base to the exp power</td>
</tr>
</tbody>
</table>
Poll in with your answer!

To go from Celsius to Fahrenheit, you multiply by 1.8 and then add 32. Which of these correctly implements this logic as a method?

A. `public static void celsiusToF(double celsius) {
    double fahrenheit = celsius * 1.8 + 32;
    return fahrenheit;
}

B. `public static void celsiusToF(double celsius) {
    double fahrenheit = celsius * 1.8 + 32;
}

C. `public static double celsiusToF(double celsius) {
    int fahrenheit = celsius * 1.8 + 32;
    return fahrenheit;
}

D. `public static double celsiusToF(double celsius) {
    return celsius * 1.8 + 32;
}
Poll in with your answer!

What value is returned from this method?

public static int returnExample() {
    for (int i = 0; i < 5; i++) {
        return i;
    }
    return -1;
}

A. -1  
B. 0  
C. 4  
D. 5
A weekly section where I introduce open problems related to our lecture topic(s) of the week.

Goals:
1. give you “conversational familiarity” with CS terminology
2. see how CS interacts with other fields and people!
3. point you in the direction of more CSE (or adjacent) classes

Note: not tested content. Just food for thought :)
We loved your C1 reflections! (1/2)

I read (skimmed?) all of your responses! Some themes:

- not previously knowing how blind people programmed
  - surprised by speed of audio & efficiency of programmer
  - debugging seemed especially challenging
- impressed by how involved he was (e.g. meeting with the VS team)
- “small things make a big difference”
Some choice quotes (1/3)

“Listening to this screen reader is painful for me and I can only imagine how difficult it was for him to get used to this path of writing code and getting this good at it. […]"

When I worked at [redacted] there was [an] initiative to improve our screen reader experience on our site. It took a lot of time and resources to make these improvements but its so important to do so. I also remember that there was a big lawsuit so a lot of websites we required to make these changes to improve accessibility […]”
We loved your C1 reflections! (2/2)

Many of you had ideas for more accessible CS!

• more accessible visual materials for blind & non-sighted users
• different types of non-keyboard inputs, especially voice input
  (stay tuned for Week 6!)
• “AI for accessibility” – a recently popular research area, but also
  with many, many challenges. Many folks at UW do research here!
Some choice quotes (2/3)

“I personally don't have a lot of knowledge when it comes to what areas of computer science are pretty accessible already, but making sure every single little thing is accessible (like the error dialogue box) seems important, but and not too difficult to implement?”
Websites are programmed with a language called HTML.

Images are represented in HTML like this:

```html
<img src="matt.png" alt="matt reading 'Dearborn' by Ghassan Zeineddine"/>
```

Let’s write a method that extracts the alt text from strings like the above! (the first step for a primitive screen reader!)
Desserts for Thought

1. Some programs take other programs as input!
   • high tea for thought: how does assignment automated testing work?
   • interested in more? Take CSE 341/401/413, or talk to Matt!

2. You can write code that supports accessibility!
   • this both involves adding context (like alt text),
     but also building accessibility tools (and UW is really good at that)
   • in section next Tuesday: building a color contrast evaluator!

3. But, automated tools can only do so much (especially on context)
Some choice quotes (3/3)

“I'll be honest, nothing resonated with me as I'm not the speaker and cannot relate to his experience regarding accessibility. In fact, I think it would be wrong of me to say I resonate with the speaker as I'm, an [abled] person, cannot speak of life and experience life as if I'm a person with disabilities.”