CSE 121 Lesson 8: Return Values

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 #cse121-8

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

• Creative Project 2 (C2) releasing later today (due Thursday May 2\textsuperscript{nd})
• R1 due tomorrow; R2 opens tomorrow (due Thursday May 2\textsuperscript{nd})
  • R2 eligible assignments: C0, P0, C1, P1
• Wednesday May 1\textsuperscript{st}: Mid-quarter Formative Feedback in class
• Quiz 0 is tomorrow in your quiz section!
  • Quiz logistics announced on Ed
• Quiz review reminders
  • Priority materials: practice quizzes, starred section problems!
  • Ed Class megathreads (great while reviewing lecture)
**Methods & Parameters**

Definition: A value passed to a method by its caller; sending information into a method

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

Calling a method with a parameter...
```
myMethod("Laufey"); // Prints out
// "Laufey is the best!"
```
(Recall) **Returns 1**

Returns allow us to send values **out of a method**

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

Calling a method that returns a value…

```java
$type$ result = myMethod(...);  // catching what is returned!
```

- Evaluates the expression
- Returns this value to where the method is called from
- Method immediately exits!
(Recall) Returns 2

Returns allow us to send values out of a method

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

Calling a method with a parameter...

```java
String s = myMethod("Laufey"); // Prints and returns
    // "Laufey is the best!"
```
(Recall) Returns 3

Returns allow us to send values out of a method

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

Calling a method with a parameter...

```java
String s = myMethod("Laufey"); // Returns
    // "Laufey is the best!"
```
## (Recall) String Methods

Usage: `<string variable> . <method>(...)`

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

String s = "bubblegum"
\[
s = s.substring(7, 8).toUpperCase() + s.substring(8) + "ball";
\]
\[
s = "g".toUpperCase() + s.substring(8) + "ball";
\]
\[
s = "G" + s.substring(8) + "ball";
\]
\[
s = "G" + "um" + "ball";
\]
## 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>
double value = 823.577564893;
double roundedValue = (double) Math.round(value * 100) / 100;

= (double) Math.round( 82357.7564893 ) / 100;

= (double) 82358.0 / 100;

= 823.58
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. ```java
public static void celsiusToF(double celsius) {
    double fahrenheit = celsius * 1.8 + 32;
    return fahrenheit;
}
```

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

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

D. ```java
public static double celsiusToF(double celsius) {
    return celsius * 1.8 + 32;
}
```
(Recall) Tricky Poll: Last line printed?

```java
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();
}
```

count: 5
COUNT: 7
COUNT: 7
count: 6
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;
}
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
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?