CSE 121 – Lesson 8

Miya Natsuhara
Autumn 2023

Music: 121 23au Lecture Tunes 🌦️

TAs: Trey Christina Sahej Vinay Kriti
     Sebastian Colton Anju Maria Minh
     Annie Janvi Jonus Shreya Vivian
     Jasmine Arkita Lydia Andy Nicole
     Christian Vidhi Luke Nicolas Simon
     Lucas Ritesh Andras Shayna Jessie
     Logan Hibbah Archit Hannah Lydia
     Jacob Julia Ayesha Aishah Yijia
Announcements, Reminders

• Programming Assignment 1 was due last night
• Creative Project 2 will be released later today
• Quiz 1 scheduled for Thursday next week, Nov 2
• Wednesday Nov 1: Mid-term Formative Feedback *in class*
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?
Metacognition: thinking about how you think
   Asking questions about your solution process

Examples

- **While debugging**: explain to yourself why you're making this change to your program
- **Before running your program**: make an explicit prediction of what you expect to see
- **When coding**: be aware of when you're not making progress, so you can take a break or try a different strategy
- **When studying**: What is the relationship of this topic to other ideas in the course?
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);
```
### (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><strong>Returns</strong> the length of the string.</td>
</tr>
<tr>
<td><code>charAt(i)</code></td>
<td><strong>Returns</strong> the character at index <code>i</code> of the string</td>
</tr>
<tr>
<td><code>indexOf(s)</code></td>
<td><strong>Returns</strong> the index of the first occurrence of <code>s</code> in the string; returns -1 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><strong>Returns</strong> 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><strong>Returns</strong> whether or not the string contains <code>s</code></td>
</tr>
<tr>
<td><code>equals(s)</code></td>
<td><strong>Returns</strong> whether or not the string is equal to <code>s</code> (case-sensitive)</td>
</tr>
<tr>
<td><code>equalsIgnoreCase(s)</code></td>
<td><strong>Returns</strong> whether or not the string is equal to <code>s</code> ignoring case</td>
</tr>
<tr>
<td><code>toUpperCase()</code></td>
<td><strong>Returns</strong> an uppercase version of the string</td>
</tr>
<tr>
<td><code>toLowerCase()</code></td>
<td><strong>Returns</strong> 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";
Example of returns: Math `class`

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

double value = 823.577564893;
double roundedValue = (double) Math.round(value * 100) / 100;
Poll in with your answer!

What is the correct implementation of a maxDatingAge method?

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

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

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

D. ```java
public static int maxDatingAge(int age) {
    return (age - 7) * 2;
}
```
Poll in with your answer!

What value is returned from this method?

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

A. 0
B. 4
C. 5
D. -1