Welcome to CSE 121!

Use this QR code as one way to ask questions!

sli.do #cse121

Simon Wu
Summer 2024

TAs:

<table>
<thead>
<tr>
<th></th>
<th>Trey</th>
<th>Hannah</th>
<th>Mia</th>
<th>Vivian</th>
<th>Jolie</th>
<th>Colton</th>
<th>Ziao</th>
</tr>
</thead>
</table>
Announcements & Reminders

• First Quiz tomorrow in your sections! See Ed reminder.
• C1 also due tomorrow, 7/10 at 11:59 pm
• Resubmission Cycle 1 due tomorrow at 11:59 pm
• P1 will be released on Friday, due next Thursday (7/17)
  • note: a big jump from C1. start early!
  • watch out for code quality – we were a little more lenient with CQ grading for the first two assignments, but will be less lenient moving forward
Last Time 1

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

```java
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);
}
```
• 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

```java
public static void myMethod() {
    /***
     * Your code here
     ***/
}
```
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.");
    }
}

A. Welcome! Glad you’re here. Hello! Glad you’re here. Goodbye!
B. Welcome! Hello! Glad you’re here. Goodbye!
C. Welcome! Hello! Goodbye!
D. Welcome! Glad you’re here. Hello! Glad you’re here. Goodbye!
(PCM) 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...

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

Returns allow us to send values out of a method

```
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...

```
<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>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 -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>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";
## 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>
Math example

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