CSE 121 – Lesson 2

Kai Daniels
Summer 2023

Music: 🌸k-pop girlies playlist🌸

sli.do #cse121
Announcements, Reminders

• **Programming Assignment 0: Code Reviews** was released Wed, due next Wed July 5\(^{th}\) 11:59 PM
• **Creative Project 0** feedback released next Tues, July 4
• Quiz 0: Monday, July 10 (take-home)
• Resubmission start next week, more on that later
Last Time...

• Expressions
  • Data types (int, double, boolean, String)
  • Operators (+, -, /, *, %)
  • Logical Operators (!, &&, ||, ==, !=, >, >=, <, <=)

• Variables
  • Container that stores a specific data type
  • Must declare & initialize!
  • Manipulate, modify, reuse

// declare AND initialize
int version = 5;
Strings and chars

- String = sequence of characters treated as one, yet can be indexed to get individual parts
- Zero-based indexing

- **Side note**: new data type! `char`, represents a single character, so we use single quotes

Strings are made up of chars!
### (PCM) 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 j 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>
Suppose s contains the String "bubble gum". Which option below would result in s containing "Gumball" instead?

String s = “bubble gum”;

A. s.substring(7) + "ball";
B. s = s.substring(7, 9) + "ball";
C. s = s.charAt(7).toUpperCase() + "ball";
D. s = 
   s.substring(7, 8).toUpperCase() 
   + s.substring(8) + "ball";
E. s = 
   s.substring(7, 8).toUpperCase() 
   + s.substring(7, 10) + "ball";
For loops are our first control structure
A syntactic structure that controls the execution of other statements.

```
for ( initialization ; test ; update ) {
    body (statements to be repeated)
}
```
(PCM) for loops!

```java
for (int counter = 1; counter <= 5; counter++) {
    System.out.println("I love CSE 121!");
}
```
(PCM) for loops!

1. Perform the initialization once at the beginning.
2. Is the test true?
   - NO: Execute the statements that are immediately after the for loop body.
   - YES: Execute the statements inside the for loop body.
3. Perform the update.
4. Execute the statements inside the for loop body.
5. If the test is false, return to step 2.
What output does the following code produce?

```java
for (int i = 1; i <= 6; i++) {
    System.out.println(i + " squared = " + i * i);
}
```

A.
- i squared = i*i
- i squared = i*i
- i squared = i*i
- i squared = i*i
- i squared = i*i
- i squared = i*i

B.
- i squared = i * i
- i squared = i * i
- i squared = i * i
- i squared = i * i
- i squared = i * i
- i squared = i * i

C.
- 1 squared = 1
- 2 squared = 4
- 3 squared = 9
- 4 squared = 16
- 5 squared = 25
- 6 squared = 36

D.
- 1 squared = 11
- 2 squared = 22
- 3 squared = 33
- 4 squared = 44
- 5 squared = 55
- 6 squared = 66
- 7 squared = 77
(PCM) String traversals

```java
// For some String s
for (int i = 0; i < s.length(); i++) {
  // do something with s.charAt(i)
}
```
Fencepost Pattern

Some task where one piece is repeated $n$ times, and another piece is repeated $n-1$ times and they alternate

g-u-m-b-a-l-l-l
Fencepost Pattern

Some task where one piece is repeated $n$ times, and another piece is repeated $n-1$ times and they alternate