CS 302: Introduction to Programming

Lecture 4
Example 1: Integer Arithmetic

Hey, explain the code this time!!! Open Eclipse ASAP~~~
Example 2: Integer Division

Alt + Tab to switch to Eclipse
Example 3: FloatingPointArithmetic

- Get started
Example 4: Powers & Roots

- Get started
Example 5: Input Example

- Martian weight converter
  - Input: Name, Weight
  - Output: Weight on Mars
Practice a program combining arithmetic and Scanner class

• Now it’s your turn
• Make a Java program that reads a number of cents and then prints out correct change, in US coinage
A Note on Arithmetic Conventions

- Often many ways to write the same thing
  - \( x = x + 1 \); vs \( x++ \); vs \( x += 1 \); vs \( ++x \);
  - \( x = x - 1 \); vs \( x-- \); vs \( x -= 1 \); vs \( --x \);
  - \( x += 5 \); vs \( x = x + 5 \);
  - \( x /= 3 \); vs \( x = x / 3 \);

```java
int x = 5, y = 5;
System.out.println(++x); // outputs 6
System.out.println(x); // outputs 6
System.out.println(y++); // outputs 5
System.out.println(y); // outputs 6
```
Strings

- Sequence of characters
- Reference type (non-primitive)
- Specified by double quotes ("")
- Can have length 0 – empty string = ""

Examples:
- String name = "Dan";
- String className = "CS302: Intro to Programming";
String Operations

- Concatenation (+)
  - Have already seen in our output statements
  - Ex: String name = "Ned" + " Stark";
  - String className = "cs";
  - int classNum = 302;
  - className = className + classNum; //className is now: “cs302”

- Length
  - String name = "Luke Skywalker";
  - int length = name.length(); //length = 14
    - Remember identifier.methodName()
Converting Strings to Numbers

- **String → int**
  - `Integer.parseInt([String])`
  - `String aNumber = "5";`
  - `int x = Integer.parseInt(aNumber);`

- **String → double**
  - `Double.parseDouble([String])`
  - `double y = Double.parseDouble("2.2");`
Chars

- Single character
- Specified by single quotes (')
- Has numeric value

Ex.

```java
char myChar = 'a';
System.out.println(myChar); //will print out: a
myChar++;
System.out.println(myChar); //will print out: b
```
### ASCII Table Values

<table>
<thead>
<tr>
<th>Dec</th>
<th>HxOctl</th>
<th>Char</th>
<th>Dec</th>
<th>HxOctl</th>
<th>Html</th>
<th>Chrs</th>
<th>Dec</th>
<th>HxOctl</th>
<th>Html</th>
<th>Chrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>NUL</td>
<td>32</td>
<td>040</td>
<td>$#32$</td>
<td>Space</td>
<td>64</td>
<td>100</td>
<td>$#64$</td>
<td>$#96$</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>SOH</td>
<td>33</td>
<td>041</td>
<td>$#33$</td>
<td>!</td>
<td>65</td>
<td>101</td>
<td>$#65$</td>
<td>$#97$</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>STX</td>
<td>34</td>
<td>042</td>
<td>$#34$</td>
<td>&quot;</td>
<td>66</td>
<td>102</td>
<td>$#66$</td>
<td>$#98$</td>
</tr>
<tr>
<td>3</td>
<td>003</td>
<td>ETX</td>
<td>35</td>
<td>043</td>
<td>$#35$</td>
<td>#</td>
<td>67</td>
<td>103</td>
<td>$#67$</td>
<td>$#99$</td>
</tr>
<tr>
<td>4</td>
<td>004</td>
<td>EOT</td>
<td>36</td>
<td>044</td>
<td>$#36$</td>
<td>$$</td>
<td>68</td>
<td>104</td>
<td>$#68$</td>
<td>$#9a$</td>
</tr>
<tr>
<td>5</td>
<td>005</td>
<td>ENQ</td>
<td>37</td>
<td>045</td>
<td>$#37$</td>
<td>$/$</td>
<td>69</td>
<td>105</td>
<td>$#69$</td>
<td>$#9b$</td>
</tr>
<tr>
<td>6</td>
<td>006</td>
<td>ACK</td>
<td>38</td>
<td>046</td>
<td>$#38$</td>
<td>$?$</td>
<td>70</td>
<td>106</td>
<td>$#70$</td>
<td>$#9c$</td>
</tr>
<tr>
<td>7</td>
<td>007</td>
<td>BEL</td>
<td>39</td>
<td>047</td>
<td>$#39$</td>
<td>$*$</td>
<td>71</td>
<td>107</td>
<td>$#71$</td>
<td>$#9d$</td>
</tr>
<tr>
<td>8</td>
<td>010</td>
<td>BS</td>
<td>40</td>
<td>050</td>
<td>$#40$</td>
<td>$!$</td>
<td>72</td>
<td>110</td>
<td>$#72$</td>
<td>$#9e$</td>
</tr>
<tr>
<td>9</td>
<td>011</td>
<td>HT</td>
<td>41</td>
<td>051</td>
<td>$#41$</td>
<td>$#$</td>
<td>73</td>
<td>111</td>
<td>$#73$</td>
<td>$#a0$</td>
</tr>
<tr>
<td>10</td>
<td>012</td>
<td>LF</td>
<td>42</td>
<td>052</td>
<td>$#42$</td>
<td>$*$</td>
<td>74</td>
<td>112</td>
<td>$#74$</td>
<td>$#a1$</td>
</tr>
<tr>
<td>11</td>
<td>013</td>
<td>VT</td>
<td>43</td>
<td>053</td>
<td>$#43$</td>
<td>$+$</td>
<td>75</td>
<td>113</td>
<td>$#75$</td>
<td>$#a2$</td>
</tr>
<tr>
<td>12</td>
<td>014</td>
<td>FF</td>
<td>44</td>
<td>054</td>
<td>$#44$</td>
<td>$-$</td>
<td>76</td>
<td>114</td>
<td>$#76$</td>
<td>$#a3$</td>
</tr>
<tr>
<td>13</td>
<td>015</td>
<td>CR</td>
<td>45</td>
<td>055</td>
<td>$#45$</td>
<td>$%$</td>
<td>77</td>
<td>115</td>
<td>$#77$</td>
<td>$#a4$</td>
</tr>
<tr>
<td>14</td>
<td>016</td>
<td>SO</td>
<td>46</td>
<td>056</td>
<td>$#46$</td>
<td>$&amp;$</td>
<td>78</td>
<td>116</td>
<td>$#78$</td>
<td>$#a5$</td>
</tr>
<tr>
<td>15</td>
<td>017</td>
<td>ST</td>
<td>47</td>
<td>057</td>
<td>$#47$</td>
<td>$#$</td>
<td>79</td>
<td>117</td>
<td>$#79$</td>
<td>$#a6$</td>
</tr>
<tr>
<td>16</td>
<td>020</td>
<td>ESC</td>
<td>48</td>
<td>059</td>
<td>$#48$</td>
<td>$_$</td>
<td>80</td>
<td>120</td>
<td>$#80$</td>
<td>$#a7$</td>
</tr>
<tr>
<td>17</td>
<td>021</td>
<td>FS</td>
<td>49</td>
<td>061</td>
<td>$#49$</td>
<td>${$</td>
<td>81</td>
<td>121</td>
<td>$#81$</td>
<td>$#a8$</td>
</tr>
<tr>
<td>18</td>
<td>022</td>
<td>GS</td>
<td>50</td>
<td>062</td>
<td>$#50$</td>
<td>$}$</td>
<td>82</td>
<td>122</td>
<td>$#82$</td>
<td>$#a9$</td>
</tr>
<tr>
<td>19</td>
<td>023</td>
<td>RS</td>
<td>51</td>
<td>063</td>
<td>$#51$</td>
<td>$#$</td>
<td>83</td>
<td>123</td>
<td>$#83$</td>
<td>$#aa$</td>
</tr>
<tr>
<td>20</td>
<td>024</td>
<td>US</td>
<td>52</td>
<td>064</td>
<td>$#52$</td>
<td>$#$</td>
<td>84</td>
<td>124</td>
<td>$#84$</td>
<td>$#ab$</td>
</tr>
<tr>
<td>21</td>
<td>025</td>
<td>Dell</td>
<td>53</td>
<td>065</td>
<td>$#53$</td>
<td>$#$</td>
<td>85</td>
<td>125</td>
<td>$#85$</td>
<td>$#ac$</td>
</tr>
<tr>
<td>22</td>
<td>026</td>
<td>Dell</td>
<td>54</td>
<td>066</td>
<td>$#54$</td>
<td>$#$</td>
<td>86</td>
<td>126</td>
<td>$#86$</td>
<td>$#ad$</td>
</tr>
<tr>
<td>23</td>
<td>027</td>
<td>Dell</td>
<td>55</td>
<td>067</td>
<td>$#55$</td>
<td>$#$</td>
<td>87</td>
<td>127</td>
<td>$#87$</td>
<td>$#ae$</td>
</tr>
<tr>
<td>24</td>
<td>028</td>
<td>Dell</td>
<td>56</td>
<td>068</td>
<td>$#56$</td>
<td>$#$</td>
<td>88</td>
<td>128</td>
<td>$#88$</td>
<td>$#af$</td>
</tr>
<tr>
<td>25</td>
<td>029</td>
<td>Dell</td>
<td>57</td>
<td>069</td>
<td>$#59$</td>
<td>$#$</td>
<td>89</td>
<td>129</td>
<td>$#89$</td>
<td>$#ba$</td>
</tr>
<tr>
<td>26</td>
<td>030</td>
<td>Dell</td>
<td>58</td>
<td>072</td>
<td>$#72$</td>
<td>$!$</td>
<td>90</td>
<td>132</td>
<td>$#90$</td>
<td>$#cc$</td>
</tr>
<tr>
<td>27</td>
<td>033</td>
<td>Dell</td>
<td>59</td>
<td>073</td>
<td>$#73$</td>
<td>$!$</td>
<td>91</td>
<td>133</td>
<td>$#91$</td>
<td>$#cd$</td>
</tr>
<tr>
<td>28</td>
<td>034</td>
<td>Dell</td>
<td>60</td>
<td>074</td>
<td>$#74$</td>
<td>$!$</td>
<td>92</td>
<td>134</td>
<td>$#92$</td>
<td>$#ce$</td>
</tr>
<tr>
<td>29</td>
<td>035</td>
<td>Dell</td>
<td>61</td>
<td>075</td>
<td>$#75$</td>
<td>$!$</td>
<td>93</td>
<td>135</td>
<td>$#93$</td>
<td>$#cf$</td>
</tr>
<tr>
<td>30</td>
<td>036</td>
<td>Dell</td>
<td>62</td>
<td>076</td>
<td>$#76$</td>
<td>$!$</td>
<td>94</td>
<td>136</td>
<td>$#94$</td>
<td>$#d0$</td>
</tr>
<tr>
<td>31</td>
<td>037</td>
<td>Dell</td>
<td>63</td>
<td>077</td>
<td>$#77$</td>
<td>$!$</td>
<td>95</td>
<td>137</td>
<td>$#95$</td>
<td>$#d1$</td>
</tr>
</tbody>
</table>

```java
int x = (int) 'a';
System.out.println(x); //output: 97
char myChar = (char) (x++);
System.out.println(myChar); //output: b
```
Method to find a specific character within a String

Strings are 0-indexed

Ex.

String name = “Bill Clinton”;
char first = name.charAt(0); //first = ‘B’
int length = name.length(); //length = ?
char last = name.charAt(length - 1); //last = ‘n’

What if I had done:
char last = name.charAt(length);
What if I want to get part of a String?

```java
StringName.substring([start], [end])
```

- Will include `charAt(start)`
- Will include `charAt(end - 1)`;
- Will NOT include `charAt(end)`
- Start, end, must be ints

Remember the 0-indexed nature of Strings

Ex.

```java
String name = "Barack Obama";
String first = name.substring(0, 3);
String last = name.substring(4);
```
Show how to actually use all the methods we learned
Reads in a social security number (SSN), formatted as XXX-XX-XXXX, where the X's represent digits. Adds 1 to that number and prints the result in the same format.
If statement

What if I want to make a decision?

Parts:

- **Boolean expression** (a statement that is either true or false)
- Code

Ex.

```java
if (5 > 1)
{
    System.out.println("Five is greater than 1");
}
```
Comparing Numbers: Relational Operations

- ==
  - Is something equal to something else
    - if (a == b)
- >
  - Greater than
- <
  - Less than
- >=
  - Greater than or equal to
- <=
  - Less than or equal to
  - !=
    - Not equal
- Precedence
  - Lower precedence than arithmetic operators
  - Ex. what does (3 + 2 < 5) evaluate to?
Comparing Strings

Do NOT use `==`  
Strings are reference variables, not primitives  
Instead use `.equals()` and `.equalsIgnoreCase()`  
Also `.compareTo()`  
  - Returns an int  
Format:  
  - `stringOne.equals(stringTwo)`

```java
String foo = "abcdef";
String bar = "ABCDEF";
if (foo.equals(bar))
{
    System.out.println("foo equals bar");
}
if (foo.equalsIgnoreCase(bar))
{
    System.out.println("foo equals bar if you ignore the case");
}
```
Else

- Code that executes if the boolean expression was false

Else

- Code under else block

Continue execution of code
String foo = "abcdef";
String bar = "ABCDEF";
if (foo.equals(bar))
{
    System.out.println("foo equals bar");
}
else
{
    System.out.println("foo doesn't equal bar");
}