- lab1 closed

Recall:

- **static methods**:

  Ex. `Math.min(6, 7);` // returns 6 
  \[\text{name of class}\]

- **Instance methods**

  Ex. `Scanner sc = new Scanner(System.in);`
  `String word;`
  `word = sc.next();` 
  \[\text{Instance of class}\]
Ex in String

- charAt(int index);

call:

String s = "happy";
char c = s.charAt(4);
↑
assigned 'y'.

- valueOf(char c)

String t = String.valueOf(c);
↑
assigned "y"
### Recall: Number Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>byte</code></td>
<td>8</td>
<td>-128 to 127</td>
</tr>
<tr>
<td><code>short</code></td>
<td>16</td>
<td>-32768 to 32767</td>
</tr>
<tr>
<td><code>char</code></td>
<td>16</td>
<td>0 ... 65535</td>
</tr>
<tr>
<td><code>int</code></td>
<td>32</td>
<td>$-2^{31}$ ... $(2^{31}-1)$</td>
</tr>
<tr>
<td><code>long</code></td>
<td>64</td>
<td>$-2^{63}$ ... $(2^{63}-1)$</td>
</tr>
</tbody>
</table>

- **Floating Point**
  - `float` 32 7 Dec. digits
  - `double` 64 14 Dec. digits

Java does arithmetic (+, -, *, /, %) only with types:
- `double` 6.0
- `float` 6.0f or 6.0F
- `long` 6L or 6L
- `int` 6
an expression like

\[ a + b \]

is evaluated as follows:

1. if \( a, b \) are of same type, then that type defines the arithmetic operation and value of expression.
2. if \( a, b \) are different types, the lower type is promoted to the higher type, which is the type of the expression.
Similar conversions take place in assignments.

Ex.

```java
int x = 6;
long y;
double z;

y = x;  // widening conversion
z = y;  // no loss of data
System.out.println(z);  // prints 6.0
```

Ex. double z = 6.5;
```java
int x = z; // narrowing conversion not allowed!
```
To do a narrowing conversion must use an explicit cast operation.

\[ \text{cast op. corresponding to type} \]

\((\text{type})\)

Ex. `Roundi.java`
Note: assignment statement

`x = y` is itself an expression with a value that can be assigned

```c
int x, y, z = 5;
```

`x = (y = z)`

value in the value assigned

Same thing:

`x = y = z;`
```java
int a, b;

a % b;  // remainder of a
       // upon division by b

14 % 6  =  2
5 % 6  =  5
100 % 60  =  40
100 / 60  =  1
100.0 / 60.0  // value 1.6666...
100.0  / 60   // same
100 / 60.0    // same
```
conditioned on:

(cond? expr : exp2)

true
false

Ex. Plural.java