Ex. CharCode1.java
Ex. CharCode2.java

**note**: "help".charAt(2)

returns 'l'

More on operators:

Recall assignment:

`Variable = Expression`

value is the value assigned.
Ex. \( \min \{ x, y, z \} = 5 \); 
\[ x = (y = 2) \]

\[ \begin{array}{ccc}
5 & 5 & 5 \\
\end{array} \]

same as:
\[ x = y = z \]

Compound assignment ops:
\[ a += b \quad \text{same as} \quad a = a + b \]
\[ a -= b \quad \quad \quad a = a - b \]
\[ a *= b \quad \quad \quad a = a \times b \]
\[ a /= b \quad \quad \quad a = a / b \]
\[ a %= b \quad \quad \quad a = a \% b \]

\( a += b \) differs from \( a = a + b \) only for integer types
Ex. \(14.0/5.0\) eval to 2.8

\[
14.0/5 \quad \text{same} \\
14/5.0
\]

\[
14/5 \quad \text{eval to 2} \\
14\%5 \quad \text{eval to 4}
\]

integer division

\[
a = (a/b)*b + (a\%b)
\]

and

\[
0 \leq (a \% b) < b
\]
Ex: HXXS.java

More shortcuts:

<table>
<thead>
<tr>
<th>Auto-increment</th>
<th>Prefix</th>
<th>Postfix</th>
</tr>
</thead>
<tbody>
<tr>
<td>++a</td>
<td>a++</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Auto-decrement</th>
<th>Prefix</th>
<th>Postfix</th>
</tr>
</thead>
<tbody>
<tr>
<td>--a</td>
<td>a--</td>
<td></td>
</tr>
</tbody>
</table>

difference is in the way (++a) and (a++) are evaluated as expressions.

++a value after increment, new
a++ value before increment, old
Ex: AutoIncrement Decrement.java

i = a b c d e f
j = 5 7 7 = 13 3
k = 6 8 6 4
l = 6

Precedence of operators:

What does this mean?
a op₁ b op₂ c

(a op₁ b) op₂ c

In case

op₁ = op₂

on
a op₁ (b op₂ c)

op₁ < op₂
Compare for equality:

\[ \text{exp1} == \text{exp2} \]

true if two expressions are identical, false otherwise.

Ternary conditional operator:

\[ \text{cond} \ ? \ \text{exp1} \ : \ \text{exp2} \]

- **Cond**: boolean expression
- **exp1**: expression evaluated if cond is true
- **exp2**: expression evaluated if cond is false
Ex. Plural.java

Read & do:

Exercise for chap 2

p. 40-44 of text
Chapter 3: Control Flow

Categories of statements:

1. Sequential: do next statement
2. Conditional (branching)
3. Iterative (looping)

Straight-line programs:
only sequential control flow

Use boolean expression in (2) \& (3)
### Relational OP
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Ex.</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>(5 &lt; 10)</td>
<td>true</td>
</tr>
<tr>
<td>(\leq)</td>
<td>(5 \leq 10)</td>
<td>true</td>
</tr>
<tr>
<td>&gt;</td>
<td>(5 &gt; 10)</td>
<td>false</td>
</tr>
<tr>
<td>(\geq)</td>
<td>(5 \geq 10)</td>
<td>false</td>
</tr>
<tr>
<td>==</td>
<td>(5 == 10)</td>
<td>false</td>
</tr>
<tr>
<td>!=</td>
<td>(5 \neq 10)</td>
<td>true</td>
</tr>
</tbody>
</table>

### Logical OP
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Ex.</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&amp;</td>
<td>((1 &lt; 2) &amp;&amp; (3 &lt; 4))</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>!((1 &lt; 2))</td>
<td>false</td>
</tr>
</tbody>
</table>
Truth Tables!

Let $A$, $B$ be boolean variables.

- $A \land B$

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>$A \land B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$F$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
</tbody>
</table>

Also called "inclusive or".

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>$A \lor B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$A$</th>
<th>$\overline{A}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
</tr>
</tbody>
</table>
English also has "exclusive or"

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A $\text{xor}$ B</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

Note: $\text{xor}$ can be created in several ways

- $(!A \& \& B) \lor (A \& \& !B)$
- $A \neq B$
Conditional Operations

- if
- if-else
- switch

bool expression

\[ \text{if (condition) \{ } \]
\[ \quad \text{true branch} \]
\[ \quad \text{true branch} \]
\[ \quad \text{true branch} \]
\[ \text{else} \]
\[ \text{true branch} \]

Note: if true branch is just one start, \( \text{else} \) are optional.
it (cond)
start
start
start

note: a set of statements in brace is called a block

if
start
start
start
start

2
also:

if (cond) {
  stmt;
  stmt;
 (stmt;)
}

? else {
  stmt;
  stmt;
(else stmt;
  stmt;)

? boolean

true branch

false branch